THE STOKES (VERSION 4) NORMAL-MODE SHALLOW WATER TRANSMISSION LOSS PROGRAM

AR-008-263

MARSHALL V. HALL

AD-A268 772

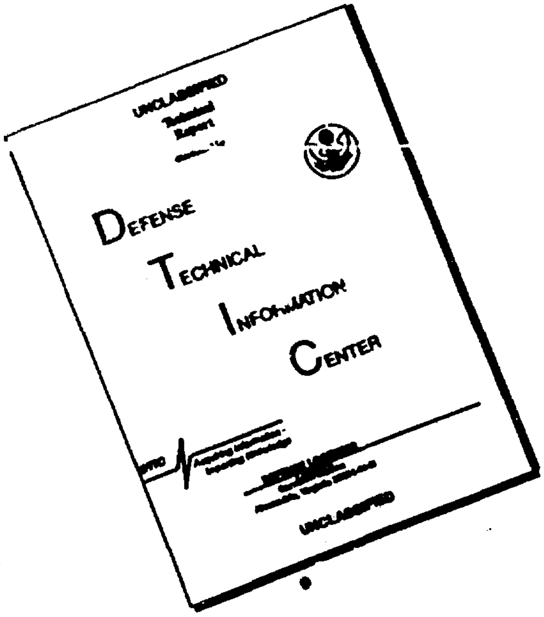
MRL COMPUTER PROGRAM: MRL-CP-1

JUNE 1993

MATERIALS RESEARCH LABORATORY

DSTO

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

The STOKES (Version 4) Normal-Mode Shallow Water Transmission Loss Program

Marshall V. Hall

MRL Computer Program
MRL-CP-1

Abstract

The STOKES program calculates the Normal-Mode-Sum solution to the acoustic wave equation for an arbitrary sound-speed profile in a water column overlying a seabed that consists of a liquid layer and a solid half-space. The water column sound-speed profile may be arbitrary except that the sound-speed adjacent to the sea-floor must not be significantly greater than that at the depths of the sound transducers. For the seabed, the user can either supply values for each geoacoustic parameter, or if the seabed is unconsolidated sediment, supply the mean grain-size. In the latter case, STOKES uses published regression equations and models to estimate the geoacoustic parameters. The mode phase-speed eigenvalues are the zeroes of a characteristic function that depends on the depth function at the sea-floor and the reflectivity of the sea-floor. STOKES calculates the mode eigenvalues to great accuracy, and is superior to SNAP if the seabed has a significant shear-speed. STOKES has been successfully run for realistic cases with bottom depths of from 60 to 120 m, and for frequencies from 32 Hz to 2 kHz. It has also performed very well when compared with bench-marks. At very low frequencies, near the cut-off frequency of the sound-speed profile as a whole (usually of the order of 10 Hz in practical cases), the imaginary part of any eigenvalue is large, and in some cases this prevents STOKES, which uses a simple root-finding procedure, from finding any mode. In such a case the Transmission Loss (at ranges of at least a few kilometres) would be exceptionally high, and its precise value would not be of practical interest.

DEPARTMENT OF DEFENCE
DSTO MATERIALS RESEARCH LABORATORY

Published by

DSTO Materials Research Laboratory
Cordite Avenue, Maribyrnong
Victoria, 3032 Australia

Telephone: (03) 246 8111
Fax: (03) 246 8999

AR No. 008-263

Contents

The STOKES (Version 4) Normal-mode Shallow Water Transmission Loss Program

1. *Overview*

1.1 The STOKES program calculates the Normal-Mode-Sum solution to the acoustic wave equation for an arbitrary sound-speed profile in a water column overlying a seabed that consists of a layer and a half-space.

1.2 As an optional extra, the Branch-Line-Integral (sometimes referred to as the "head wave" or the "continuous mode contribution") is also included, although the algorithm in the current version is valid only for an iso-speed water column.

1.3 The water column profile is specified by the depths (DZ) and sound-speeds (C) at a user-specified number of knee-points in the profile. The final knee-point is at the depth of the sea-floor (the water/seabed interface). The vertical interval between each pair of neighbouring knee-points is referred to as a layer. The number of layers (denoted by NWAT) is of course 1 fewer than the number of knee-points. Each layer is assumed to have flat (horizontal) and smooth boundaries.

1.4 The seabed is modelled by a homogeneous liquid layer overlying a homogeneous solid (or liquid) half-space. If appropriate, the thickness of the layer (variable THICK_L) can be set to zero.

1.5 The main differences between the versions of STOKES are the formulas used to estimate the shear-speed of the seabed. There are also small differences in the estimated compressional speed of the seabed. The sources of the various formulas used are listed in Annex A of this manual.

In addition, STOKES version 3 was the first version to allow the seabed to be described as a (liquid) layer over a solid or liquid half-space (the previous versions allowed only for a homogeneous half-space)

Version 4 requires the input data file to present the data in NAMELIST format.

1.6 STOKES does not include the effect of the Scholte wave that, at frequencies up to around 10 Hz, can be significant at depths within a wavelength of the sea-floor.

2. STOKES' Novel Features

2.1 Geoacoustic Profiles

STOKES offers the user the option of supplying only the "Hamilton" mean grain-size (HMGS) of the sediment, as defined in Hamilton and Bachman [1982]. (HMGS is the average of the 16th, 50th and 84th percentile phi-values if available, otherwise it is the average of the 20th, 50th, and 70th percentiles). From the specified value of HMGS, STOKES employs Hamilton/Bachman regression equations to determine a "package" of the required acoustic properties of unconsolidated sediment. The user also has the option of ignoring this package and using independent values for the seabed's acoustic parameters.

2.2 Complex Arithmetic

STOKES uses complex arithmetic while determining the mode eigenvalues (in the complex plane) whether the seabed is liquid or solid, while allowing in either case for an arbitrary sound-speed profile in the water column. Other programs (such as SNAP) use a first-order perturbation method to estimate the effect of either seabed absorption or seabed rigidity on the imaginary part of the mode eigenvalues, and neglect their effects on the real part altogether. The possible error in this technique can be significant, especially if the shear speed of the seabed is comparable with the sound-speed in the water (since the damping of such modes is high). An example of STOKES' superior accuracy to that of SNAP is given in Annex B.

2.3 Branch-Line Integral

Unless the variable Ans_bli is set to 'N' (in the input data file), STOKES will include the Branch-Line Integral (BLI). The algorithm is adapted from Brekhovskikh (1980, p. 335) and will be accurate only if the water column is iso-speed. The BLI can be significant at frequencies near the cut-off frequency of the first normal mode.

3. Model of the Sound-Speed Profile

The Water Column

The sound-speeds at the top and bottom of the N'th layer are denoted by CT(N) and CB(N) respectively.

3.1 In any water layer for which CT equals CB, the program assumes the sound-speed to be constant. The corresponding dependence of the acoustic depth function with depth (for each mode) is a linear combination of the sine and

cosine functions.

3.2 In any water layer for which CT is unequal to CB, the program assumes the reciprocal square sound-speed to be a linear function of depth. The corresponding equation for the depth-dependent component of the wave function (Z) may be expressed as a Stokes Equation (hence the name of the program, although several existing programs also make use of this feature).
The acoustic depth eigenfunction (for each mode) is a linear combination of the two types of Airy function.

The dimension of arrays that depend on the number of layers is specified by the parameter NIM, whose current value is 20. Because of the limitations in the accuracy with which Airy functions can be calculated, it has been found advisable to keep the number of layers in which they are needed to no more than 4, and lower if possible. For example, if the initial number of "Airy" layers is greater than 4, and/or the values of CT and CB in some layers differ by only 0.1 or 0.2 m/s, then the most suitable layers should be converted into iso-speed layers.

The Seabed

3.3 The seabed is modelled by a homogeneous liquid sediment layer overlying a homogeneous half-space, each of whose properties is specified by the user. The layer can be "deleted" either by setting THICK_L = 0, or by setting its acoustic properties equal to those of the halfspace (if the latter is also a liquid). In the program, parameters of these two components are distinguished by the post-scripts "_L" and "_H" respectively.

3.4 The primary variables for specifying the sediment layer (if THICK_L > 0) and the half-space are DUMMY_L and DUMMY_H respectively.

3.5 If DUMMY_L is less than 1000 then it is taken to be the Phi-value (PHI) of the sediment layer HMGS. Empirical expressions are then used to express Density, compressional sound speed and attenuation in terms of PHI.
If not known directly, the PHI value may be estimated from the list of Sediment Types given in Table 1 of this manual. The sources of the regression equations used by the program to estimate the above parameters are as follows:

(a) Density RHO_L is calculated from the porosity using a modified version of Bachman's (1985) regression equation for porosity. The modification is to force the porosity to level off at 0.35 as HMGS approaches 0 phi, rather than to approach 0.21 as predicted by Bachman's equation. (In v. 3, Bachman's (1985) direct regression equation for density was used, but this is regarded as predicting too high a density for HMGS < 2 phi.)

(b) Compressional sound-speed CPR_L (Bachman, 1989);

(c) The damping coefficient of the compressional wave, CAYP_L, is calculated using the regression equations presented by Hamilton (1972).

If DUMMY_H is less than 1000 then it is taken to be the Phi-value (PHI) of the sediment half-space HMGS. RHO_H, CPR_H, and CAYP_H are obtained using the same empirical expressions as used in the layer. Since the half-space can be a solid, its shear speed and absorption need to be determined:

(d) The shear sound-speed CSR_H is set equal to its value at the depth of one-tenth of the compressional wavelength, using the depth-dependence of shear modulus observed by Iwasaki and Tatsuoka (1977). A brief explanation of this procedure is contained in Annex A.

(e) The damping coefficient of the shear wave, CAYS_H, is calculated using the algorithm presented by Stoll (1989, pages 96 and 122-123).

3.6 If DUMMY_L is greater than 1000, then it is treated as the layer's compressional sound-speed (CPR_L). In this case, the user needs to include data specifying values for the density (RHO_L), and the attenuation coefficient of the compressional wave (CAYP_L). This flexibility is provided to allow for cases where data more accurate than the above-mentioned regression equations are available.

If DUMMY_H is greater than 1000, then it is treated as the half-space's compressional sound-speed (CPR_H). In this case, the user needs to include data that specify values for the density (RHO_H), shear-sound speed (CSR_H), and attenuation coefficients for both the compressional (CAYP_H) and shear waves (CAYS_H). This flexibility is provided to allow for cases where data more accurate than the above-mentioned regression equations are available.

4. *The Program and its Subprograms*

4.1 The master source code of the STOKES program is in the DSTO-Sydney VAX 11/780; directory [HALL.SHALLOW.STOKES]. The author may be contacted on this computer by electronic mail to the following address:

hall@dstosy.dsto.gov.au

A glossary of the variables (or parameters) as used in the STOKES program is contained in Annex C.

Software Drivers

The software driver used by the master is the NAG(TM) FORTRAN Library (REAL*8 Implementation). The programs can be linked without a NAG library, and will run providing the parameters (namely the mode group-speed and the Branch-Line-Integral) that depend on NAG routines are avoided. They are avoided by setting the variables Ans_group and Ans_bli (in namelist STOKES_IN) both equal to 'N'.

NAG is available from:

The Numerical Algorithms Group Ltd	Phone: +44 865 511245
Wilkinson House	Fax: +44 865 310139
Jordan Hill Road	Contact: Robert Marrell
Oxford OX2 8DR UK	

STOKES Version 4 does not itself produce any "plot" data files, although there are two associated stand-alone plotting programs that read the output data files that are produced. These plotting programs ("Plotmode" and "Plottlos") use the GRAFKIT(TM) Graphics Application.

GRAFKIT (TM) is available from:

SCO, Inc.
740C. South Pierce Avenue
Suite 15
Louisville Colorado 80027-9989 USA

Phone: +1 303 666 5400
Fax: +1 303 666 7054
Contact: LaNell Svoboda

The Input Peripheral used for STOKES is a standard VDU such as the VT220. Output Peripherals used are a standard line printer, a model LN03P or a Postscript laser-writer. If the associated plotting programs are run, then the same output peripherals may again be used, or a graphics terminal VT240 can be used to inspect screen plots if hard copies are not required.

4.2 The NAG Library is included for the following purposes:

(a) If Ans_group = 'Y', then subroutine GROUP_SPEED is called, which in turn calls the NAG integration routine D01AJF (REAL*8 Implementation) during calculation of the modes' Group Speeds. If the Group Speeds are not required then setting Ans_group equal to 'N' will avoid the call to the NAG routine. Further details are given in the description in section 4.6 below of the NORMGROUP and GROUP_SPEED source codes.

(b) Unless Ans_bli = 'N', subroutine RELINT calls the NAG complex function S15DDF (complex*16 implementation), which is related to the complex complementary error function.

In each of the above cases, calls to the NAG subprograms can of course be replaced by calls to equivalent functions in other FORTRAN libraries.

Individual Subprograms

The LINK command for STOKES is:

Link MAIN, elastic_props, MODES, FNCTNS, CHAREQ, NORMGROU, RELINT, AIRYRYAN, NAG/L

(inclusion of "NAG/L" is optional, as discussed elsewhere, although its omission will of course cause warning messages to appear).

A convenient way to recall these file names for the Link command is to precede it with a DIR .OBJ command to list the Object modules. The main program and the subroutines all have statements INCLUDE 'PARAM.FOR' and INCLUDE 'COMMON.FOR'. PARAM.FOR is a short file that contains PARAMETER and TYPE statements; while COMMON.FOR is a short file that contains COMMON and DATA statements. The source codes of the main program and the subprograms, except for subroutine AIRY, are listed in Annex D.

For the associated plotting programs, the Link commands are:

Gk2load plotmode; and Gk2load plottlos

This manual is in 9 files (*.man) within the above directory.

4.3 After entering RUN MAIN, the user is prompted to enter the input data file name. MAIN creates a corresponding output data file (its name is 'Z_' concatenated to the start of the input data file name). From the input data file, MAIN reads data from namelists DATA_IN, STOKES_IN, LAYER_IN, and HALF_IN. If the geo-acoustic parameters are to be calculated (as described in section 3.5) then subroutine ELASTIC_PROPS is called. MAIN next calls subroutine MODES, which reads further data from namelist MODE_IN (although in most cases these variables can be left at their default values). This call is followed by calls to NORMGROU, FNCTNS, and RELINT (described below).

4.4 File MODES.FOR contains SUBROUTINE MODES, which calculates the (complex) Eigenvalues of the locatable acoustic normal modes {V(1), V(2),...}. MODES gives the user the option of either supplying estimates of mode Phase Speeds or letting the program search for the modes (via variable ANS_MANUAL in namelist MODE_IN).

The dimension of arrays that depend on the number of modes is specified by parameter MIM, whose current value is 300.

4.41 In the "search" option, each mode's Phase-Speed is sought, starting from a threshold minimum phase speed Vmin, which is calculated as described in the section below on numerical problems. (The reason for determining a minimum phase-speed is also described in the section on numerical problems). MODES steps along in small (positive) increments in phase speed (V), calling subroutine CHAREQ (see below) to evaluate the Characteristic Function (CF), the zeroes of which are the mode eigenvalues. When the real part of CF changes sign, MODES uses an iteration process to determine the exact eigenvalue. The characteristic function must reduce to a small value (TOLERANCE) during the iteration that finds the eigenvalues.The default value for TOLERANCE is 5.E-4 * 32 /FREQ. It is made to increase as frequency decreases because at low frequencies the imaginary part is large so the error in the complex trigonometric or Airy function is also large.

In addition, TOLERANCE is used as the threshold for checking whether a search has converged to an eigenvalue already found. At high frequencies it is found that modes are liable to have very close eigenvalues.

TOLERANCE may need to be increased if NWAT is large, or to locate additional higher modes if the default value produces too few modes - as evidenced by the highest damping rate not being large enough to render the next mode negligible.

4.42 In the "supply" option (ANS_MANUAL = 'Y'), the user supplies the complex phase-speed e.g. (1600,10) for mode 1 (etc) via variable VEST in namelist MODE_IN. This option is useful at very low frequencies, in particular near the cut-off frequency of the shallow-water wave-guide, where the imaginary part (damping) of the eigenvalue is so large that it lies outside the catchment of the stepping search. The complex phase-speed V(1) (for example) at a low frequency can be estimated by using the "search" option to determine it at higher frequencies and extrapolating the trend to the lower frequency of interest.

4.43 MODES writes messages to the screen to indicate how many modes have been located, and whether the run has been aborted (numerical overflows are the usual cause). If the run aborts, the user should re-RUN MAIN and, for the

number of modes (variable MAXIN in namelist MODE_IN), enter the number that were/was successfully located during the first run (unless it was zero, in which case the user enters the number of modes for which Phase-speed estimates can be supplied in accordance with the procedure described in sect. 4.42).

4.5 File CHAREQ.FOR contains subroutine CHAREQ, which calculates CF for each trial value of the wavenumber eigenvalue (XI). If Airy functions are required, then subroutine AIRY (which is contained in AIRYRYAN.FOR) is called. This subroutine was written by Dr Frank. J. Ryan, while he was with Theoretical Physics Division, Science Applications Inc., La Jolla, California 92037, USA. The NAG library also has complex Airy functions, but their running time is significantly longer than Ryan's subroutine.

CHAREQ also calculates the coefficients WA and WB that will be used later to specify the depth functions UR and US.

4.6 File NORMGROU.FOR contains Subroutines NORMGROUP and GROUP_SPEED; and Functions RFUNCT and HFUNCT. For each mode, NORMGROUP calculates the normalization factor WN (in order that the resulting mode sum gives the correct "weight" to each mode). The algorithm used for the contribution of a solid half-space to WN is adapted from Eq. (6) of Ellis and Chapman (1985), which is based on Eq. (7) of Koch et al (1983). If Ans_group = 'Y', NORMGROUP calls subroutine GROUP_SPEED. The algorithm used for the contribution of a solid half-space to group speed is adapted from Eq. (9) of Koch *et al.* (1983). The calculation of the group speed requires a function (whose real and imaginary parts are RFUNCT and HFUNCT respectively) to be integrated over each fluid layer; in the present configuration this is done using the NAG routine D01AJF.

The statements that make use of the NAG routine D01AJF are the following:

```
CALL D01AJF(RFUNCT, 0.D0, THICK, EPSABS, EPSREL, RINTEGRAL,
:     ABSERR, WORK, LW, IW, LIW, IFAIL)
CALL D01AJF(HFUNCT, 0.D0, THICK, EPSABS, EPSREL, HINTEGRAL,
:     ABSERR, WORK, LW, IW, LIW, IFAIL)
DENOM = DENOM + DCMPLX(RINTEGRAL, HINTEGRAL) * DEL
     /CT(N)**2
```

The parameter ABSERR is determined by the NAG routine. IFAIL is revalued (from its initial value of 0) if there is a failure.

4.7 File FNCTNS.FOR contains subroutine FNCTNS, which for each mode calculates the depth eigenfunction US at the source depth (SOURCE); and the eigenfunctions UR at the receiver depths. The Receiver depths are as follows: minimum (DTMIN), maximum (DTMAX), and increment (DELDT). The number of receivers is therefore (DTMAX - DTMIN)/DELDT + 1. The dimension of arrays that depend on number of receiver depths is specified by parameter JIM, whose current value is 101.

FNCTNS writes the results to a binary file to be read by an auxiliary plotting program, if required.

4.8 File RELINT.FOR contains subroutine RELINT, which calculates:

(1) the sum of the Normal Modes; and

(2) (unless Ans_bli = 'N') the Branch-Line Integral ("head wave" or lateral arrival). This algorithm, presented in Brekhovskikh (1980, section 37),

calls the NAG complex*16 function S15DDF, which is related to the complex complementary error function.

These components are added to give the Relative Intensity (R_I) from the source to each receiver depth, as a function of horizontal range.
RELINT will crash if any of the ranges is zero, since R_I there is infinite.

The dimension of arrays that depend on the number of ranges is specified by parameter KIM, whose current value is 480.

The operation - 10 * log (base 10) is then performed on R_I to convert them into Transmission Loss in decibels (dB) before being printed. Since the R_I value at the surface is zero (it is a pressure release surface), RELINT will crash if any of the source or receiver depths is zero.

If variable ANS_RI (in namelist STOKES_IN) is set to anything other than ' ', then RELINT writes the Transmission Losses to a binary file that can be read by program PLOTTLOS. Otherwise it writes them to the output file Z_....

Debugging Facilities

4.9 There is a debugging facility in that the user can cause additional information to be written to the output file from within 3 of the subroutines by giving a value (of 1) to particular variables. The subroutines are MODES, CHAREQ, and NORMGROUP, and the corresponding debug variables are MODE_RITE, CH_RITE, and NORM_RITE respectively.

Running STOKES on a DOS PC

4.10 The adjustments that need to be made to the VAX version of STOKES, in order that it can run on a DOS PC, are listed in Annex E. These changes are needed primarily because of the limited memory available in DOS, and also because the structure of directories on a PC will generally be different from that on a VAX.

5. *Data Entry and an Example*

5.1 The four NAMELIST blocks in the main program are as follows:

```
namelist / data_in / nwat, dz, ctort, thick_L,
:       rmax, delr, freq, SOURCE, dtmin, dtmax, deldt, rho_w
namelist / stokes_in / ans_mode_FN, ans_ri, ANS_GROUP, ANS_BLI,
:       NORM_RITE
namelist / Layer_in / rho_L, dummy_L, cayp_L
namelist / half_in / rho_h, dummy_h, csr_h, cayp_h, cays_h
```

The NAMELIST block in Subroutine MODES is:

```
namelist / mode_in / ans_manual, maxin, MODE_RITE, TOLERANCE,
:           vest, ch_rite
```

5.2 An example of an input data file for a sound signal at a frequency of 100 Hz transmitting within a 1-layer profile is shown in Table 2. A table of values for Transmission Loss will be written to the output data file, which is shown in Table 3.

5.3 The following comments should be read in conjunction with Table 2:

(1) The final break-point depth in the water column sound-speed profile is the depth of the sea-floor.

(2) Horizontal ranges are in KILO-units: maximum (RMAX), and increment (DELR).

Table 1: *Wentworth scale of sediment mean grain-size*

Sand type:	VC	C	M	F	VF
Grain-size (mm):	1.4	0.71	0.35	0.18	0.09
Phi:	-0.5	+0.5	1.5	2.5	3.5
Silt type:	C	M	F	VF	
Grain-size (micron):	44	22	11	6	
Phi:	4.5	5.5	6.5	7.5	

in which the letters have the following meanings:

C - coarse; F - fine; M - medium; V - very.

Table 2: Example of an input data file for the STOKES (v. 4) program

```
[HALL.SHALLOW.STOKES]summer.DAT
Features of this example:
(1) the water layer's SSP has a -ve gradient (CB < CT)
(2) the sediment layer's thickness is zero
(3) the half-space is characterized by its Phi value

 &DATA_IN
 FREQ = 100
 NWAT = 1
 rho_w = 1025  ! Optional. If not supplied, rho_w is calculated from CTORT
 DZ =    0     100
 CTORT = 1540  1520
 THICK_L = 0
 source = 20
 DTMIN = 20
 DTMAX = 80
 DELDT = 20
 RMAX = 30
 DELR = 2
 &

 &stokes_in
 ans_mode_fn = ' '
 ans_group = 'y'
 norm_rite = 0
 ans_ri = ' '
 &

 &layer_in
 dummy_L = 0    !since thick_L = 0, dummy_L would not be read in this example
 &

 &half_in
 dummy_h = 3    ! since Dummy_h < 1000 in this example, it is treated as the
                mean grain size in phi-units, and the geoacoustic parameters
                will be calculated usi g regression equations.
 &

 &MODE_IN
 ans_manual = 'N'
 ch_rite = 0
 MAXIN = 100
 mode_rite = 0
 tolerance = 0
 vest = (1550., 1.)     ! since ans_manual = 'N', vest would not be read in
                        this example
 &
```

Table 3: The output data file corresponding to the input file shown in Table 2

```
Input Data File:  STOKES summer      Cpr_h: 1735 Freq: 100

LAYER NUMBER          :      1        2
LAYER-TOP DEPTHS(m):       0      100
Layer-top Speeds    :  1540.00
Layer-bot Speeds    :  1520.00
G - Coefficients    :  3.5E-02
Water Density (Kg/m3):      1025.
Layer thickness (m):        0.
Half-space Density (Kg/m3): 1915.  "Dummy":        3.0
Half-space CP (m/s):    1734.578  31.496    Shear-speed (m/s):  142.70  1.87
Half-space Absorptions (dB/Km/Hz):     0.571     5.025
Approx. Cut-off Frequency:       8. Hz
No. of modes:  7

 MODE        PHASE           LOSS/KM      NORMALIZATION        GROUP     SKIP
NUMBER       SPEED                                             SPEED    DISTANCE
  1    1531.5547  1.1E-01    0.249     4.39E+01 -1.70E+00    1529.1
  2    1544.8448  1.8E-01    0.401     3.30E+00 -6.42E-02    1526.4     1780
  3    1564.9983  3.2E-01    0.718     1.07E+00 -1.35E-02    1515.9     1199
  4    1594.9321  5.3E-01    1.133     5.43E-01 -5.71E-03    1498.8      833
  5    1636.4557  8.5E-01    1.728     3.32E-01 -3.78E-03    1476.1      628
  6    1691.9488  1.6E+00    3.007     2.27E-01 -5.05E-03    1454.9      498
  7    1765.4741  5.4E+00    9.446     1.59E-01 -1.20E-02    1356.3      405

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

Estimated rate of absorption at     100 Hz ,for temperature of  27 deg. C
is 9.0E-04 dB/Km

At R(1) and DT(1), BLI =  1.3E-12;  Mode Sum =  1.4E-03

No. of receivers:   4                       No. of ranges:   15

                TRANSMISSION LOSSES (dB re m^2)

                    Source & Receiver Depths (m)
   20.0      20.0      40.0      60.0      80.0

 RANGE (Km)
   2.000     56.9      58.6      52.4      57.3
   4.000     58.5      61.4      60.0      66.3
   6.000     58.3      65.3      64.5      69.1
   8.000     72.9      67.8      66.1      63.0
  10.000     66.3      67.8      77.3      65.0
  12.000     66.8      67.5      70.3      72.7
  14.000     75.9      68.0      70.1      73.9
  16.000     75.9      69.1      72.2      76.0
  18.000     70.9      71.1      82.2      81.5
  20.000     76.1      74.1      81.9      74.1
  22.000     83.1      77.9      76.8      74.4
  24.000     75.7      81.4      79.6      76.8
  26.000     82.2      81.8      82.3      76.7
  28.000     82.1      80.5      88.1      78.7
  30.000     78.1      79.9      88.6      86.4

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

6. Numerical Problems That May Arise

6.1 Deep-water Transmission

The STOKES program is unsuitable for sound-speed profiles in which the sound-speed at great depth approaches its surface value, i.e. profiles which could support convergence-zone transmission. The reason for this is that the low-order modes would have negligible amplitude at great depth (whether or not there is an isothermal surface mixed layer); and numerical limitations of the computer cause the characteristic function to be calculated with insufficient accuracy for the location of eigenvalues (because for STOKES, the characteristic function is a function of the reflectivity of the sea-floor).

6.2 Thick Surface Mixed Layer

6.21 The following problem is liable to occur if the program is run at frequencies higher than around 1 kHz. As the surface mixed-layer thickness (MLT) increases, (the number of trapped modes in the surface duct also increases), the amplitude of the first trapped mode may become exponentially small near the base of the duct and in numerical models this can give rise to a numerical exponent underflow problem. (In determining mode eigen values, these amplitudes have to be evaluated regardless of whether the user specifies that any of the transducers are near the base of the layer.) This problem is circumvented by limiting the interval of sound speed in which modes are sought; but such a procedure means that some low-order modes (i.e. modes whose phase-speeds are close to the minimum sound-speed in the profile) may not be located. This omission will usually be unimportant providing either the source or receiver is in or near the surface mixed layer, since low order HF modes generally have negligible amplitude within the surface layer.

6.22 In profiles that contain more than one layer, setting Vmin equal to the minimum sound-speed of the profile (the usual procedure, to ensure that the first mode is not omitted) may cause some arguments of the Airy functions that are calculated to be large enough to cause numeric overflow. These arguments, which in the program are called ZT (for the Top of each layer) and ZB (for the Bottom of the layer) are given by

```
ZETA = (PI * FREQ /GRAD)^(2/3) * (C^2/V^2 - 1)          (1)
```

where

GRAD is the sound-speed gradient in the layer;
C is the sound-speed at the top (or bottom) of the layer; and
V is the trial phase-speed of the mode.

For the VAX computer, the maximum value of ZETA is ZMAX = 25.56. If larger values were encountered, then program execution would abort and the message

ERROR IN NAG ROUTINE IFAIL = 1

would be written at the terminal. This problem is liable to occur if there is a surface isothermal mixed-layer (since in such a layer the sound-speed gradient is only 0.016 /s), and if V < C. For the above case, for example, if the initial trial value of V were CMIN = 1525 (m/s), then in the first layer

GRAD = (1540 - 1539.8) /12.5 = 0.016

so that

ZETA = (PI * 400 /0.016)^(2/3) * (1540^2/1525^2 - 1)
= 36.3

For this case, program execution would therefore abort. The solution that has been embodied in the program is to consider only phase speeds greater than the threshold speed (VMIN) that corresponds to ZETA = ZMAX. VMIN is obtained by inverting Eq. (1) to express V in terms of C, FREQ, GRAD and ZMAX. If the value of VMIN thus calculated is less than the minimum of the sound-speed profile (CMIN) then it is re-set to equal CMIN. (A Phase Speed less than CMIN would cause a run to abort). The partial mode sum obtained by the above method includes all the modes that are significant, providing either the source or receiver is in or near the surface mixed layer. If both transducers are deep however, then this method is liable to omit some significant modes.

7. *References*

Bachman, R.T. (1985).
Acoustic and physical property relationships in marine sediment, *Journal of the Acoustical Society of America*, **78**, pp. 616-621.

Bachman, R.T. (1989).
Estimating velocity ratio in marine sediment, *Journal of the Acoustical Society of America*, **86**, pp. 2029-2032.

Brekhovskikh, L.M. (1980).
Waves in layered media (second ed.). New York: Academic Press.

Bryan, G.M. and Stoll, R.D. (1988).
The dynamic shear modulus of marine sediments, *Journal of the Acoustical Society of America*, **83**, pp. 2159-2164.

Chapman, D.M.F., Ward, P.D. and Ellis, D.D. (1989).
The effective depth of a Pekeris waveguide, including shear wave effects, *Journal of the Acoustical Society of America*, **85**, pp. 648-653.

Ellis, D.D. and Chapman, D.M.F. (1985).
A simple shallow water propagation model including shear wave effects, *Journal of the Acoustical Society of America*, **78**, pp. 2087-2095.

Hall, M.V. (1992).
Reflectivity of sand sea-beds at a frequency of 10 Hz, *Acoustics Australia* (special issue on Underwater Acoustics), **20** (3), pp. 91-94.

Hamilton, E.L. (1972).
Compressional-wave attenuation in marine sediments, *Geophysics*, **37**, pp. 620-645.

Hamilton, E.L. (1976).
Shear-wave velocity versus depth in marine sediments: a review, *Geophysics*, **41**, pp. 985-996.

Hamilton, E.L. (1978).
Sound velocity-density relations in sea-floor sediments and rocks, *Journal of the Acoustical Society of America*, **63**, pp. 366-377.

Hamilton, E.L. (1980).
Geoacoustic modeling of the sea floor, *Journal of the Acoustical Society of America*, **68**, pp. 1313-1340.

Hamilton, E.L. and Bachman, R.T. (1982).
Sound velocity and related properties of marine sediments, *Journal of the Acoustical Society of America*, **72**, pp. 1891-1904.

Iwasaki, T. and Tatsuoka, F. (1977).
Effects of grain size and grading on dynamic shear moduli of sands, *Soils and Foundations*, **17** (3), pp. 19-35.

Jensen, F.B. and Ferla, M.C. (1979).
SNAP: the SACLANCTEN normal-mode acoustic propagation model (Memorandum SM-121). La Spezia: SACLANT Undersea Research Centre.

Koch, R.A., Penland, C., Vidmar, P.J. and Hawker, K.E. (1983).
On the calculation of normal mode group velocity and attenuation, *Journal of the Acoustical Society of America*, **73**, pp. 820-825.

Mellen, R.H., Scheifele, P.M. and Browning, D.G. (1987).
Global model for sound propagation in sea water Part II: Geosecs pH data analysis (TR 7925). New London: Naval Undersea Systems Center.

Stoll, R.D. (1989).
Sediment acoustics. Berlin: Springer-Verlag.

Yamamoto, T. (1983).
Acoustic propagation in the ocean with a poro-elastic bottom, *Journal of the Acoustical Society of America*, **73**, pp. 1587-1596.

Annex A

Formulas Used for Estimating the Geoacoustic Parameters in the Various Versions of STOKES

The geoacoustic parameters are the density, the compressional and shear speeds, and the compressional and shear absorption coefficients (dB/Hz/km) of the seabed.

(1) Version 1 (1986)

The seabed was modelled as one homogeneous solid half-space. The Compressional speed was provided by the user. Density was then calculated from the equation presented in the caption to Fig. 2 of Hamilton (1978), and Shear speed was calculated from Eqs. (4) to (7) of Hamilton (1980).

(2) Version 2 (1987)

The user supplied the mean-grain-size of the sediment (on the assumption that it was unconsolidated granular sediment) and the geoacoustic parameters were obtained as follows.

Density

Density was calculated using the density versus mean grain size regression equation for the continental terrace presented in Table 1 of Bachman (1985):

RHO = 2380 - 173 * PHI + 7 * PHI**2

Compressional speed

From the data for CPR and PHI presented in Tables 1 and 2 of Hamilton and Bachman (1982), the following 3-term regression equation was fitted, using the standard least-squares method, and giving equal weight to the average CPR at each of the 9 sediment types:

CPR_v2 /CB = 1.246 - 0.0439 * PHI + 0.00162 * PHI**2

where CB is the sound-speed of the bottom water.

Compressional absorption coefficient

CAYP was obtained from the set of regression equations, in which the mean grain size is the independent variable, in the caption to Fig. 3 of Hamilton (1972).

Shear-speed

Shear-speed

In an attempt to allow for the depth-dependence of the shear-speed CSR, it was calculated at a depth of one half of a compressional wavelength. From Eqs. (14) and (16) of Hamilton (1976), CSR was expressed as CSR = A * DEPTH**B, whereA and B are given as per the following table:

PHI	0.5	2.9
A	127	120
B	0.31	0.28

In the absence of additional information at that time, both A and B were assumed to be linear functions of PHI. The resulting expression for CSR was

CSR = (129 - 2.9 * PHI) * (CPR_v2 /2 /FREQ)**(0.316 - PHI /80)

(3) Version 3 (1990)

The layer could be defined by specifying either the mean-grain-size (in phi-units) or the compressional sound-speed, density, and compressional absorption coefficient.

The half-space could be defined by specifying either the mean-grain-size (in phi-units) or the compressional and shear sound-speeds, density, and the compressional and shear absorption coefficients.

If the mean-grain-size was specified, then the acoustic parameters were determined as follows:

Density

Density was calculated using the same equation as for v.2

Compressional Speed

Compressional speed was calculated from the regression equation presented in Bachman (1989):

CPR_v3 /CB = 1.296 - 0.0601 * PHI + 0.00283 * PHI**2

The CPR_v2 and CPR_v3 curves intersect at PHI = 4.8 and 8.6, and gradually diverge as PHI either decreases below 4.8 (at PHI = 0, CPR_v2 is 4% less than CPR_v3) or increases above 8.6. Between the two intersections, the maximum difference occurs at PHI = 6.7, where CPR_v2 is 0.42% greater than CPR_v3. These differences between the two curves are mainly attributed to the fact that the CPR_v2 regression equation was not weighted by the number of samples, which tended to be less at the smaller PHI values. Although small, these differences can cause perturbations in the bottom reflectivity at small grazing angles, and hence significant differences in the calculated Relative Intensity at ranges sufficiently great that the damping of even Mode No. 1 is significant.

Shear Speed

Shear Speed was calculated from the empirical expression for shear modulus presented by Bryan and Stoll (1988) in terms of effective pressure and porosity. The porosity was obtained from the regression equation given in Bachman (1985), and the effective pressure (at the layer/half-space interface) was obtained from the difference in the densities of the layer and the bottom water. The v.2 and v.3 values for CSR are very similar at PHI = 0; but the v.3 value approaches zero more rapidly as the PHI value increases. At PHI = 5, for example, the v.3 value for CSR is about 50% of the v.2 value. Since the ratio CSR /CPR is small (<10%), these differences cause only a small perturbation in the bottom reflectivity, and this perturbation is usually most noticeable at medium grazing angles. The change in CSR does not alter the calculated Transmission Loss significantly at ranges sufficiently great that only the first mode is significant. There may, however, be some variation in the interference pattern at shorter ranges where high order modes are still significant.

Following Hamilton (1980), the shear absorption coefficient (cays) was set equal to 13.

(4) Version 4 (1993)

If the mean-grain-size is specified, then the geoacoustic parameters are calculated as follows:

Density

rho = porosity * rho_w + (1 - porosity) * rho_s
where
rho_s = 2660 * quartz + 2710 * calcite
in which
calcite = 0.3 (a rough average than should be replaced by a more accurate value if known),
quartz = 1 - calcite
and porosity is given by Bachman's (1985) regression equation for MGS > 2.5:
porosity = 0.208 + 9.43 e-2 * aphi - 3.34 e-3 * aphi**2;
For aphi < 2.5, poros is made higher than Bachman's eq. (see his Fig.3)
if (aphi .lt. 2.5) poros = 0.35 + (aphi - 1) /1.5 * (.423 - .35)
(at aphi = 2.5, Bachman's poros = 0.423)
if (aphi .lt. 1.) poros = 0.35
(Bachman's expression equals 0.21 at MGS = 0 phi, which is too low a value)

Compressional speed: as for Version 3.

Compressional Absorption Coefficient: as for Versions 2 and 3.

Shear speed

The shear modulus of the granular frame is expressed as G = L * Pe^pee, where real L is given by Iwasaki and Tatsuoka (1977) [for their minimum strain of 1.e-6], and Pe is the effective stress at a depth of one-tenth of a compressional wavelength. Providing Cayp is i.a.w. Hamilton (1972), and G is i.a.w. Iwasaki and Tatsuoka (1977), tests have shown that defining the homogeneous seabed to have an effective shear-speed calculated at this depth causes the reflectivity (as a function of grazing angle) to be very close to the exact reflectivity calculated by taking into account the coupling between the shear and compressional waves in the sediment. A description of these calculations is presented by Hall (1992).

Shear absorption coefficient

The algorithm presented by Stoll (1989, pages 96 and 122-123) has been adopted. This algorithm is based on the Biot theory of sound transmission through a saturated granular medium.

Annex B

Eigenvalue Accuracies: Comparison of STOKES with other Normal-Mode Models

Chapman, Ward and Ellis (1989) presented the phase speeds of the trapped modes as calculated by three normal-mode models, SHEAR2, DODGE, and SNAP (Jensen and Ferla, 1979), for each of two shallow water (loss-less) sound-speed profiles (labelled "B" and "E"). The SHEAR2 model was regarded as being exact, since it makes no approximation in incorporating the effects of shear waves and absorption in the seabed. The values calculated by SHEAR2 are therefore regarded as the bench-mark. DODGE is an approximate normal-mode model based on the effective depth concept.

In both profiles, the bottom depth was 100 m, and the water column had a constant sound-speed of 1500 m/s. The respective geo-acoustic profiles were as follows (the compressional and shear absorption coefficients are both zero):

Seabed	Rho	Cpr	Cpi	Csr	Csi
"B"	1746	1700	0	200	0
"E"	2054	2150	0	650	0

For seabed "B", the real parts of the phase-speeds (the imaginary parts were not reported) of the 6 trapped modes for a frequency of 100 Hz were:

Mode	SHEAR2	DODGE	SNAP	STOKES
1	1503.632	1503.628	1503.594	1503.632
2	1514.73	1514.67	1514.59	1514.730
3	1533.90	1533.64	1533.65	1533.903
4	1562.19	1561.42	1561.84	1562.192
5	1601.13	1599.47	1600.74	1601.129
6	1652.67	1649.99	1652.38	1652.689

For seabed "E", the real parts of the phase-speeds of the 10 trapped modes for a frequency of 100 Hz were:

Mode	SHEAR2	DODGE	SNAP	STOKES
1	1504.019	1504.024	1503.726	1504.019
2	1516.25	1516.29	1515.11	1516.247
3	1537.21	1537.43	1534.81	1537.211
4	1567.88	1568.55	1563.95	1567.878
5	1609.81	1611.50	1604.31	1609.803
6	1665.42	1669.11	1658.59	1665.420
7	1738.56	1745.89	1730.74	1738.555
8	1835.35	1849.17	1827.37	1835.343
9	1965.84	1991.76	1959.09	1965.837
10	2142.35	---	2141.06	----*

* STOKES did not find Mode 10 in this case. STOKES' search method sometimes cannot find modes whose phase speeds are very close to CPR_H, because in the vicinity of CPR_H the characteristic function (the zeroes of which are the eigenvalues) is a complicated and rapidly varying function of the phase-speed.

It can be seen that the STOKES results are in much better agreement with the bench-mark than are either the SNAP or DODGE results.

The errors in the SNAP calculated phase speeds were attributed to SNAP's treatment of the lower half-space as a fluid, rather than a solid, medium. The SNAP mode phase speeds therefore do not depend on the values of the shear speed (or acoustic absorption either). The errors in SNAP are larger for the seabed (E) that has the higher shear speed.

Annex C

Glossary of Stokes Variables

ANS_BLI	The Branch-Line Integral (in RELINT) is included unless ANS_BLI = 'N'. See sections 2.3, 4.1, 4.2, 4.8, 5.1.
ANS_MANUAL	If set to 'Y', then subroutine MODES will expect MAXIN approximate eigenvalues (VEST) to be supplied. This optional "manual" method of supplying initial eigenvalues may be necessary at very low frequencies (near the cut-off frequency of the shallow water channel). See sections: 4.4, 4.42, 5.1.
ANS_MODE_FN	Key for writing mode amplitudes vs depth to a binary file (to facilitate plotting). If ANS_MODE_FN = ' ', then mode amplitudes are not written. See section 5.1.
ANS_GROUP	Group-speeds of the modes are calculated if ANS_GROUP = 'Y'. See sections: 4.1, 4.2(a), 4.6, 5.1.
ANS_RI	Key for writing Transmission Loss results to a binary file (to facilitate plotting). If ANS_RI = ' ', then only an Ascii file is produced. See sections: 4.8, 5.1.
CAYP_L & CAYP_H	Absorption coefficient of compressional (P) wave in the seabed [dB/km/Hz]: _L - layer; _H - half-space. If the P absorption is given as say p dB per cycle, then CAYP = 1000 * p /CPR. See sections: 3.5, 3.6, 5.1, Annex A.
CAYS_H	Absorption coefficient of shear (S) wave in the half-space [dB/km/Hz]. If the S absorption is given as s dB per cycle, then CAYS_H = 1000 * s /CSR_H. See sections: 3.5(e), 3.6, 5.1, Annex A.
CH_RITE	Debug data from subroutine CHAREQ are written if CH_RITE = 1 See sections: 4.9, 5.1.
CPR_L & CPR_H	Compressional sound-speed [m/s]: _L - layer; _H - half-space. See sections: 3.5, 3.6, Annex A.
CSR_H	Shear sound-speed in half-space [m/s]. See sections: 3.5, 3.6, 5.1, Annex A.
CTORT	Either CT [m/s] (if > 1000), or Temperature [deg. C] (if < 1000) See section 5.1.
DELR	Incremental horizontal range [km]. See sections: 5.1, 5.3
DZ	Array of depths of the water column Sound-Speed Profile [m] See sections: 1.3, 5.1.

DTMIN	First target (receiver) depth [m]. See section 4.7.
DTMAX	Final target (receiver) depth [m]. See section 4.7.
DELDT	Incremental target (receiver) depth [m]. See section 4.7.
DUMMY_L	If < 1000, this is treated as the mean grain size (in phi-units) of the sediment in the Layer, and CPR_L, RHO_L, and CAYP_L will be calculated using regression equations. If > 1000, it is treated as CPR_L, and the user will need to supply RHO_L and CAYP_L (in NAMELIST LAYER_IN). See sections 3.4, 3.5, 3.6, 5.1.
DUMMY_H	As for DUMMY_L, but for the Half-space rather than the Layer. If > 1000, the user should supply RHO_H, CAYP_H, CSR_H and CAYS_H (in NAMELIST HALF_IN). See sections 3.4, 3.5, 3.6, 5.1.
FREQ	Acoustic frequency [Hz]. See section 6.22.
JIM	Dimension of arrays that depend on number of receiver depths (current value is 101). See section 4.7.
KIM	Dimension of arrays that depend on number of range points
LW	Size of working space for NAG integration routine in NORMGROUP (current value is 80). See section 4.6.
LIW	Size of auxiliary array for NAG integration routine (LIW = LW/8 + 2)
MAXIN	Maximum number of modes to be included. Default value is MIM See sections 4.43, 5.1.
MIM	Dimension of arrays that depend on number of modes (current value is 300). See section 4.4.
MODE_RITE	Debug data from subroutine MODES are written if MODE_RITE = 1. See sections 4.9, 5.1.
NIM	Dimension of arrays that depend on number of layers in the water column (current value is 20). See section 3.2.
NORM_RITE	Debug data from subroutine NORMGROUP are written if NORM_RITE = 1. See sections 4.9, 5.1.
NWAT	Number of layers in the water column's Sound-Speed Profile. See sections 1.3, 5.1.
RHO_W	Density of the water column [kg/m^3]. See sections 5.1, Annex A.

RHO_L, RHO_H	Densities of the sediment layer and the half-space [kg/m^3]. See sections 3.5, 3.6, 5.1.
RMAX	Maximum horizontal range [km]. See sections 5.1, 5.3.
SOURCE	Source depth [m]. See sections 4.7, 5.1.
THICK_L	Thickness of the sediment layer [m]. See sections 1.4, 3.3, 3.4, 5.1.
TOLERANCE	The (small) value that the characteristic function must reduce to during the iteration in MODES in order that an eigenvalue is determined to have been located. Tolerance is also used by MODES as the threshold for checking whether a search has converged to an eigenvalue already found. See sections 4.41, 5.1.
VEST	Array of approximate mode complex phase-speed [m/s] eigenvalues that need to be supplied if ANS_MANUAL = 'Y'. See sections 4.42, 5.1.

Annex D

Source Code of the STOKES Program and Subprograms

Contents:

1. MAIN.FOR (Program MAIN)

2. PARAM.FOR and COMMON.FOR ('include' files)

3. ELASTIC_PROPS.FOR (Subroutines ELASTIC_PROPS and UPPERCASE, and Functions BER and BEI)

4. MODES.FOR (Subroutine MODES)

5. CHAREQ.FOR (Subroutine CHAREQ)

6. FNCTNS.FOR (Subroutines FNCTNS and POTENTIAL)

7. NORMGROU.FOR (Subroutines NORMGROUP and GROUP_SPEED, and Functions RFUNCT and HFUNCT

8. RELINT.FOR (Subroutine RELINT)

```
c       File:  [hall.shallow.stokes]  main.for


        PROGRAM MAIN                                ! Version  4  (1993)
C                  Author: Marshall Hall, DSTO Sydney
C           COPYRIGHT       COMMONWEALTH OF AUSTRALIA    1993
C   Calculates Normal Mode solution for NWAT "STOKES" water layers in shallow
C  water over a 2 homogeneous-layer seabed of which the upper layer is liquid
C  and the lower half-space is solid or liquid
C                         ------
C
C   INDICES ARE USED AS FOLLOWS: M - modes, N - layers

C  The LINK command is either (using Ryan's Airy routine):
C     Link  MAIN,elastic_props,MODES,FNCTNS,CHAREQ,NORMGROU,RELINT,AIRYRYAN,NAG/L
C    or (using the NAG Airy routines):
C     Link  MAIN,elastic_props,MODES,FNCTNS,CHAREQ,NORMGROU,RELINT,AIRYNAG,NAG/L
C     The NAG library is needed for
C  (1) kelvin functions for Biot viscosity (for imag G) - used by Elastic_props
C  (2) integration routine D01AJF used for
C   calculating mode group speed in Subroutine NORMGROUP. This can, of course,
C   be replaced by an alternative integration routine. FFI see comments within
C   the NORMGROU source code)
C  (3) complex error function, for calculation of BLI in subroutine RELINT

         include 'param.for'
         REAL CTORT(NIM),  CBORT(NIM)
         INCLUDE 'COMMON.FOR'
        DATA WI /(0.D0, 1.D0)/  LEN_STR / 12 /
        DATA FNAME /'STOKES                Cpr_h:        Freq:      '/
         namelist / data_in / freq, nwat, rho_w, dz, ctort, thick_1,
       :   SOURCE, dtmin, dtmax, deldt, rmax, delr
         namelist / stokes_in / ans_mode_FN, ans_ri, ANS_GROUP, ans_bli,
      :            NORM_RITE
         namelist / Layer_in / rho_l, dummy_l, cayp_l
         namelist / half_in / rho_h, dummy_h, csr_h, cayp_h, cays_h, hs_h
         write (*,*) 'This is STOKES version 4.0 (dated 5 Jan 93)'
        write (*,*)
C  ask the user for the input data file name
         WRITE (*,*) 'Enter the input data filename (up to 12 characters):'
         READ (*,105) FNAME(7+1:7+LEN_STR)
 105    FORMAT (A12)
          OPEN (UNIT = JIN, file=FNAME(7+1:7+LEN_STR), STATUS='OLD')
C  A corresponding output data file is created ('Z_' is concatenated to the start
C  of the input data file name):
         OPEN (UNIT = JOUT, file = 'Z_'//FNAME(7+1:7+LEN_STR),
      :                      STATUS = 'unknown')
        ALOGT = LOG(10.)
        PI = 4 * DATAN (1.D0)
        RA_TO_D = 180 /PI
        WRON = 1 /PI
C  READ  INPUT SOUND-SPEED PROFILE, freq, ranges, and sensor depths:
         READ (JIN, data_in)
         IF (FREQ .LE. 0.) THEN
          WRITE (*,*) 'Frequency le 0'
          STOP
           END IF
        OMEGA = 2 * PI * FREQ
            WRITE (FNAME(37:40), 145) NINT(FREQ)
 145    FORMAT (I4)

C  Calculate sea surface temperature from surface sound speed:
         IF (CTORT(1) .GT. 1000.) THEN
         TEMP = 12.6 +
        : 0.287 * (CTORT(1) - 1500.) + 2.12E-3 * (CTORT(1) - 1500.)**2
        ELSE
         TEMP = CTORT(1)
        END IF
```

```
c       File:  [hall.shallow.stokes]  main.for


C    Calculate  useful  properties  of  NWAT  water  layers:
         DO 200 N = 1, NWAT
            IF  (CTORT(N)  .LT.  1000.)  THEN
             CT(N)  = 1522 + 2.756 *  (CTORT(N)  - 20)  -
     :       0.0388  *  (CTORT(N)  - 20)**2  + 0.0163  * DZ(N)
          ELSE
             CT(N)  =  CTORT(N)
          END IF
  200  CONTINUE
         DO 250 N = 1, NWAT
          CBORT(N)  =  CTORT(N+1)
            IF  (CBORT(N)  .LT.  1000.)  THEN
             CB(N)  = 1522 + 2.756 *  (CBORT(N)  - 20)  -
     :       0.0388  *  (CBORT(N)  - 20)**2  + 0.0163  * DZ(N+1)
          ELSE
             CB(N)  =  CBORT(N)
          END IF
D           WRITE  (*,*)  'Ct(1):', CT(1), '     Cb(1):', CB(1)
          IF  (CB(N)  .EQ.  CT(N))  GO  TO  250
          CKSQ_T  =  (OMEGA  /CT(N))**2
          CKSQ_B  =  (OMEGA  /CB(N))**2
          GCUBE  =  (CKSQ_B  -  CKSQ_T)  /(DZ(N+1)  -  DZ(N))
          IF  (GCUBE  .GT.  0.)  GEE(N)  =     EXP  (LOG  (GCUBE)/3)
          IF  (GCUBE  .LT.  0.)  GEE(N)  =  -  EXP  (LOG  (-GCUBE)/3)
  250  CONTINUE
         .cbottom  =  cb(nwat)
          IF  (RHO_W  .LE.  0.)  THEN
            CB_AMEND  =  cbottom  -  0.0163  *  DZ(NWAT+1)
            TEMP_B  =  12.6  +  0.287  *  (CB_AMEND  -  1500)  +
     :                  2.12E-3  *  (CB_AMEND  -  1500)**2
            RHO_W  =  1028.33  -  0.081482  *  TEMP_B  -  0.004938  *  TEMP_B**2
         END  IF
C  read  the  options:  ans_mode_FN,  ans_ri,  ANS_GROUP,  ans_bli,  NORM_RITE
          READ  (JIN,  stokes_in)
           call  UPPERCASE(ans_bli)
           call  UPPERCASE(ANS_ri)
            call  UPPERCASE(ANS_mode_fn)
           call  UPPERCASE(ANS_group)

C  The  seabed:  Enter  the  properties  of  the  layer  and  the  half-space:
          IF  (THICK_L  .EQ.  0.)  GO  TO  300

C   The  shear  modulus  of  the  layer  is  set  to  zero  (the  layer  MUST  be  a  liquid)

          READ  (JIN,  Layer_in)
          if  (dummy_l  .lt.  1000.)  then
           aphi  =  dummy_l
              call  elastic_props(aphi,  cbottom,  rho_L,  cpr_L,  cayp_L,  'L',
     :                               csr_L,  cays_L)
        else
           cpr_L  =  dummy_L
             if  (rho_L  .eq.  0.)  rho_L  =  density(cbottom,  rho_w,  cpr_L)
        end  if
C   complex  sound-speed  in  the  layer:
            CPI  =  CPR_L**2  *  CAYP_L  *  ALOGT  /PI  /4.E4
            WCP_L  =  CMPLX  (CPR_L,  CPI)

  300  READ  (JIN,  half_in)
          if  (dummy_h  .lt.  1000.)  then
           aphi  =  dummy_h
              call  elastic_props(aphi,  cbottom,  rho_h,  cpr_h,  cayp_h,  'H',
     :                               csr_h,  cays_h)
        else
           cpr_h  =  dummy_h
             if  (rho_h  .eq.  0.)  rho_h  =  density(cbottom,  rho_w,  cpr_h)
            if  (cays_h  .eq.  0.  .and.  csr_h  .gt.  0.)
     :                     cays_h  =  1000  *  hs_h  /csr_h
        end  if
```

```
c       File:  [hall.shallow.stokes]  main.for


C    complex  sound-speeds  in  the  half-space:
          CPI  =  CPR_H**2  *  CAYP_H  *  ALOGT  /PI  /4.E4
          WCP_H  =  CMPLX  (CPR_H,  CPI)
          CSI  =  CSR_H**2  *  CAYS_H  *  ALOGT  /PI  /4.E4
          WCS_H  =  CMPLX  (CSR_H,  CSI)
          WRITE  (FNAME(27:30),  145)  NINT(cpr_h)

        IF  (THICK_L  .GT.  0.)  THEN             !  include  seabed  layer  in  DZ
         NLAY  =  NWAT  +  1
          DZ(NLAY+1)  =  DZ(NLAY)  +  THICK_L
          ct(nlay)  =  real(wcp_L)
          cb(nlay)  =  real(wcp_L)
        ELSE
         NLAY  =  NWAT
        END  IF

         WRITE  (JOUT,205)  FNAME
  205   FORMAT  ('  Input  Data  File:    ',  A40/)
        WRITE  (JOUT,215)   (    N  ,  N  =  1,  NWAT+1)
  215  FORMAT  ('  LAYER  NUMBER          :',  I7,  7I9)
         WRITE  (JOUT,225)   (INT(DZ(N)),  N  =  1,  NWAT+1)
  225   FORMAT  ('  LAYER-TOP  DEPTHS(m):',  I7,  7I9)
         WRITE  (JOUT,235)   (CT(N),  N  =  1,  NWAT)
  235   FORMAT  ('  Layer-top  Speeds     :',  8F9.2)
         WRITE  (JOUT,245)   (CB(N),  N  =  1,  NWAT)
  245   FORMAT  ('  Layer-bot  Speeds     :',  8F9.2)
         WRITE  (JOUT,255)   (GEE(N),  N  =  1,  NWAT)
  255   FORMAT  ('  G  -  Coefficients     :',  1p  8  E9.1)
         WRITE  (JOUT,265)  RHO_W
  265   FORMAT  ('  Water  Density  (Kg/m3):',  F10.0)
         WRITE  (JOUT,  275)  THICK_L
  275   FORMAT  ('  Layer  thickness  (m):',  F9.0)
C     Layer     "_L"
        IF  (THICK_L  .GT.  0.)  THEN
          WRITE  (JOUT,295)  RHO_L,  dummy_l
  295     FORMAT  (  '  Layer  Density  (Kg/m3):',  f10.0,  '    "Dummy":',  f10.1)
          WRITE  (JOUT,  305)    WCP_L,  CAYP_L
  305     FORMAT  (  '  Layer  Compressional  speed  (m/s):    ',  2F9.3,
     :                '  ;  Absorption:',  f6.3)
        END  IF
C     Half-space     "_H"
             WRITE  (JOUT,325)  RHO_H,  dummy_h
  325      FORMAT  ('  Half-space  Density  (Kg/m3):',f6.0,  '    "Dummy":',f10.1)
         WRITE  (JOUT,  335)    WCP_H,  WCS_H
  335   FORMAT  (  '  Half-space  CP  (m/s):',  F12.3,  F8.3,
     :  '       Shear-speed  (m/s):  ',  F7.2,  F6.2)
         WRITE  (JOUT,345)  CAYP_H,  CAYS_H
  345   FORMAT  ('  Half-space  Absorptions  (dB/Km/Hz):',  2F9.3)
         if  (csr_h  .lt.  cbottom)  then
          if  (cpr_h  .lt.  cbottom)  then
             freq_cut  =  cbottom  /4  /DZ(NLAY+1)  /SQRT(1  -  (cpr_h  /cbottom)**2)
        else
             freq_cut  =  cbottom  /4  /DZ(NLAY+1)  /SQRT(1  -  (cbottom  /CPR_H)**2)
        end  if
       else
           freq_cut  =  cbottom  /4  /DZ(NLAY+1)  /SQRT(1  -  (cbottom  /CSR_H)**2)
       end  if
         WRITE  (JOUT,355)  freq_cut
  355   FORMAT  ('  Approx.  Cut-off  Frequency:',  F8.0,  '  Hz')

C    Subroutine  "MODES"  calculates  the  (complex)  eigenvalue  of  each  mode:
         CALL  MODES  (MAXOUT)
         WRITE  (JOUT,395)  MAXOUT
  395   FORMAT  ('  No.  of  modes:',  i3)
         if  (maxout  .eq.  0  .and.  ans_bli  .eq.  'N')  then
          write  (*,*)  'Since  no  mode  has  been  found  and  the  Branch  Line
       :  Integral  was  not  requested,  program  will  now  stop'
          write  (*,*)  'Have  you  tried  entering  an  approximate  mode  phase
```

```
     : speed? (Ans_manual needs to be set to ''y'')'
      stop
     end if
      WRITE (*,*) ' Calling NORMGROUP'
      WRITE (*,*)

      DO MODE = 1, MAXOUT
        CALL NORMGROUP(ANS_GROUP, NORM_RITE)
      END DO

      WRITE (JOUT,*)
      WRITE (JOUT,405)
405   FORMAT ('    MODE',7X,'PHASE',9X, 'LOSS/KM        NORMALIZATION',
     :        '          GROUP      SKIP')
      WRITE (JOUT, 415)
415   FORMAT ( ' NUMBER',6X,'SPEED', 41X, 'SPEED      DISTANCE')
      IF (MAXOUT .GE. 1) THEN
      M = 1
       ELPK = - 2.D4 /ALOGT * DIMAG(WIG(M))
          WRITE (JOUT,625) 1, OMEGA /WIG(M), ELPK, WN(M), GROUP(M)

      DO 600 M = 2, MAXOUT
       ELPK = - 2.D4 /ALOGT * DIMAG(WIG(M))
          WRITE (JOUT,625) M, OMEGA /WIG(M), ELPK, WN(M), GROUP(M),
     :                    - INT (2 * PI /(WIG(M) - WIG(M-1)))
600   CONTINUE
625   FORMAT (I4, F12.4, 1p e9.1, 0p F9.3, 2X, 1p 2E10.2, 0P f9.1, i9)
      END IF
      WRITE (JOUT,*)

      IF (SOURCE .LT. 0.) THEN
        WRITE (*,*) 'Source depth < 0'
        STOP
          END IF
       WRITE (*,*) 'Calling FNCTNS'
       NTRANS = nint((DTMAX - DTMIN) /DELDT) + 1
      DO J = 1, NTRANS
      DT(J) = DTMIN + (J - 1) * DELDT
      END DO
        CALL FNCTNS (MAXOUT, SOURCE, NTRANS, Ans_MODE_FN)
      WRITE (JOUT,855)
      IF (DTMIN .LE. 0.) THEN
        WRITE (*,*) 'First Receiver depth le 0'
        STOP
          END IF
      IF (RMax .LE. 0.) THEN
        WRITE (*,*) 'Maximum Range le 0'
        STOP
          END IF

       WRITE (*,*) 'Calling RELINT'
        CALL RELINT (MAXOUT, SOURCE, NTRANS, RMAX, DELR, ans_bli, Ans_RI)
      CLOSE (UNIT = JIN)
      WRITE (JOUT,855)
855   FORMAT (/39(' _'))
      STOP
      END
```

```
C     file  [hall.shallow.stokes]  param.for
          PARAMETER (NIM = 20, MIM = 300, JIM = 101)
          IMPLICIT COMPLEX*16 (U - Z)
          REAL*8 CT, CTI, CB, CBI, PI, WRON, ZED
          integer ch_rite
            CHARACTER  ans_bli,  Ans_MODE_FN,  Ans_RI,  ANS_GROUP,  fname*40
```

```
C     File:  [hall.shallow.stokes]common.for
          COMMON / CONSTANT/ ALOGT, PI, RA_TO_D, WI, WRON
          COMMON / EIGEN / WIG(MIM), WN(MIM), GROUP(MIM)
          COMMON / FUNCT / WA(NIM,MIM), WB(NIM,MIM), US(MIM), UR(JIM, MIM)
          COMMON / IDENTIFY/ OMEGA, DT(JIM), NLAY
          COMMON / M_AND_N / MODE, N, wdeln(nim), wcost_h(mim), wcosg_h(mim)
          COMMON / NAME / FNAME, FREQ, freq_cut, TEMP
          COMMON / SEABED/ THICK_L, WCP_L, RHO_L, WCP_H, WCS_H, RHO_H
          COMMON / WATER/ DZ(NIM), CT(NIM), CB(NIM), GEE(NIM), RHO_W, NWAT
        DATA JIN / 10 /        JOUT / 11 /
```

```
c    File:  [hall.shallow.stokes]  elastic_props.for



          subroutine  ELASTIC_PROPS(aphi,  cbot,rho,cpr,cayp,bed_key,csr,cays)

        include 'param.for'
        real viscous, void
        REAL*8 BER, BEI, KAPPA
        EXTERNAL BER, BEI
          REAL*8  DER_R(14),  DER_I(14),  EREST_R(14),  EREST_I(14)
        character bed_key
        INCLUDE 'COMMON.FOR'
       data GRAV / 9.8 /
c proportion  of  calcite  in  the  solid  phase  (replace  by  accurate  value  if  known):
       data calcite / 0.3 /
       DATA  VISCOUS /  1.E-3 /            struct / 1.25 /
C Bachman's  (1989)  regression  for  speed  ratio  in  terms  of  aphi  (Eq.  7):
        ratio = 1.296 - 0.0601 * aphi + 0.00283 * aphi**2
        if (ratio .gt. 1.21) ratio = 1.21
       cpr = cbot * ratio

C   Bachman's  (1985)  regression  equation  for  porosity  in  terms  of  aphi:
        poros = 0.208 + 9.43 e-2 * aphi - 3.34 e-3 * aphi**2
C for  aphi  < 2.5,  poros  is  made  higher  than  Bachman's  eq.  (see  his  Fig.3)
c   poros(2.5)  =  0.423
        if (aphi  .lt.  2.5)   poros = 0.35 + (aphi - 1) /1.5 * (.423 - .35)
        if (aphi  .lt.  1.)    poros = 0.35
         write (*,*)  ' Porosity (%):',  nint(100 * poros)
       quartz = 1 - calcite
       RHO_S = 2660 * quartz + 2710 * calcite
       rho = poros * rho_w + (1 - poros) * rho_s

C   Attenuation  coefficients  (Hamilton,  1972,  fig.  3):
                                CAYP = 0.4556 + 0.0245 * aphi
        IF (aphi .GT. 2.6) CAYP = 0.1978 + 0.1245 * aphi
        IF (aphi .GT. 4.5) CAYP = 8.0399 - 2.5228 * aphi +
    :                                                   0.20098 * aphi**2
        IF (aphi .GT. 6.)   CAYP = 0.9431 - 0.2041 * aphi +
    :                                                  1.17 E-2 * aphi**2

C   SHEAR  MODULUS
        if (bed_key .ne. 'H') return  !  shear  speed  not  calculated  for  Layer
C   shear  modulus  of  "damp"  frame  is  expressed  as  G = L * Pe^pee,  where  real  L  is
C   given  by  Iwasaki  and  Tatsuoka  (1977)   with  strain  of  1.e-6:
       VOID = POROS /(1 - POROS)
        fn_void = (2.17 - void)**2 /(1 + void)      !  Hardin & Black  expression
        if (fn_void .lt. 0.) fn_void = 0.
      pee = 0.4
       ell = 900 * (grav * 1.e4)**(1 - pee) * fn_void

C   Stoll  (1989)  p.  96:
      alfa = 0.2
       alfa_comp = 1 - alfa
       ratio = freq /1.e4
      arg = pi /2. * alfa
        fnr = ratio**alfa_comp * sin(arg) + ratio**(2 * alfa_comp)
        fni = ratio**alfa_comp * cos(arg)
        den = 1 + 2 * ratio**alfa_comp * sin(arg) + ratio**(2 * alfa_comp)
       decre_vlf = 0.03
        if (aphi .gt. 2.) decre_vlf = 0.015 * aphi         ! Stoll (1989) p. 122
C   weighting  of  freq-dependent  term  increases  with  phi-value
      ex = aphi /6
       DECRE_damp = (decre_vlf + ex * pi * fni /6 /den) /
    :                           (1 + ex * fnr /6 /den)
        WELL_damp = ell * CMPLX(1., DECRE_damp /3.14159)
C   Biot  (1956b):
       diam = 0.001 /2.**aphi
```

```
c    File:  [hall.shallow.stokes]  elastic_props.for

        s_nort = 6 /diam
        kozeny = 5
         PERMEAB = 1. /s_nort**2 * POROS**3 /(1 - POROS)**2 /kozeny
         SIZE = 2 * sqrt(kozeny * permeab /poros)
         KAPPA = SIZE * SQRT(OMEGA * RHO_W /VISCOUS)
         IF (KAPPA .GT. 50.)THEN
           WFUNCT = KAPPA /4 * DCMPLX(1.D0,1.D0)/SQRT(2.)
        END IF
         IF (KAPPA .EQ. 0.) THEN
         WFUNCT = 1.
        END IF
         IF (KAPPA .GT. 0. .AND. KAPPA .LE. 50.) THEN
         IFAIL = 0
Calculate derivative of real and Imag parts of Biot's modified Kelvin function
           CALL D04AAF (KAPPA, 1, 1.D-3, DER_R, EREST_R, BER, IFAIL)
         IFAIL = 0
           CALL D04AAF (KAPPA, 1, 1.D-3, DER_I, EREST_I, BEI, IFAIL)
             WTBIOT = DCMPLX(DER_R(1),DER_I(1)) /DCMPLX(BER(KAPPA),BEI(KAPPA))
           WFUNCT = KAPPA * WTBIOT /4 /(1 - 2 * WTBIOT /WI /KAPPA)   !Biot eq. (3.19)
        END IF
C   Terms in Yamamato's (1983) expressions for the fast compressional & shear
C   speeds:
            WEMDASH = STRUCT * RHO_W /POROS - WI * VISCOUS * WFUNCT /OMEGA
     :                                          /PERMEAB
         WIVIDE_S = 1 - RHO_W**2 /WEMDASH /RHO
         WELL_WET = WELL_damp /WIVIDE_S
c the following is roughly correct providing Cayp is iaw Hamilton (1972), and
c   G is iaw Iwasaki and Tatsuoka (1977):
         depth_equiv = 0.1 * cpr /freq
         p_e = (rho - rho_w) * grav * depth_equiv
         WGEE = WELL_wet * P_E**pee
        wcs = sqrt (wgee /rho)
        csr = real(wcs)
C   CSI = CSR_H**2 * CAYS_H * ALOGT /PI /4.E4
         cays = dimag(wcs) /csr**2 /alogt * pi * 4.e4
        return
        end


C the following 2 functions are called by subroutine ELASTIC_PROPS:

         REAL*8 FUNCTION BER(X)
        REAL*8 X, S19AAF
        IFAIL = 0
         BER = S19AAF(X, IFAIL)
        RETURN
        END


         REAL*8 FUNCTION BEI(X)
        REAL*8 X, S19ABF
        IFAIL = 0
         BEI = S19ABF(X, IFAIL)
        RETURN
        END
```

```
c    File:  [hall.shallow.stokes]  elastic_props.for


         SUBROUTINE  uppercase(string)
         CHARACTER  string*(*)
         INTEGER  i,l,it,ia,iz,iua
        ia=ICHAR('a')                 ! ascii codes
         iz=ICHAR('z')
         iua=ICHAR('A')

        l = LEN(string)
        DO 1 i=1,l
            it = ICHAR(string(i:i))
              IF  (it.GE.ia.AND.it.LE.iz)  string(i:i)  =  CHAR(it-ia+iua)
      1 CONTINUE
        RETURN
        END


         function density (cwr, rho_w, cpr)
C                Author: Marshall Hall, DSTO Sydney
C         COPYRIGHT      COMMONWEALTH OF AUSTRALIA    1992
C For  sediment  whose  dilational  speed  Cp  is  known,  e.g.  from
C refraction  profiling:  (1)  uses  Hamilton's  (1978)  eqns  (cemented  sediment),  or
C   Bachman's  (1989)  equation  (uncemented  sediment)  to  get  density,
C  (2)  calculates  porosity  from  density

C           This  subprogram  is  in  [HALL.SHALLOW.stokes]elastic_props.for

         character  ans_sed
          real  cw,  temp_b,  rho_w,  cp,  calcite,  quartz,  clay
         real  eta,  rho_g,  poros,  cayp
          REAL*8  bachman,  rho_doub        !  double  precision  counterpart  to  Density
         integer  ifail
         EXTERNAL  bachman
         common  /  richard  /  cw,  cp
         data  calcite  /  0.3  /     clay  /  0.1  /       ans_sed  /  'U'  /
         quartz  =  1  -  calcite  -  clay
         RHO_g  =  quartz  *  2650  +  calcite  *  2710  +  clay  *  2720

        cw = cwr     !  so  that  Cw  and  Cp  can  be  passed  from  argument  of  subroutine
        cp = cpr     !   to  Function  Bachman  via  the  common  block  ("richard")

         call  uppercase(ans_sed)
        IFAIL = 0
         if  (ans_sed  .eq.  'U')  then
c  for  a  given  cp,  the  density  of  the  medium  is  obtained  by  inverting  Bachman89:
           CALL  C05ADF(1100.d0,  2500.d0,  1.d-5,  0.d0,  bachman,  rho_doub,
     :                                    ifail)
          density  =  rho_doub
        else
          if  (cp  .ge.  3000.)  then              !  Hamilton  (1978),  Fig.  2
           density  =  1979  +  0.112  *  cp
         else
            density  =  2351  -  7497  /(cp  /1000)**4.656
         end  if
        end  if
        return
        end
```

```
c    File:  [hall.shallow.stokes]  elastic_props.for



        real*8  function  bachman  (rho)
        implicit  none
        real*8  rho
        real  cw,  cp
        common  /  richard  /  cw,  cp
c   Bachman  (1989),  Eq.  (9):
        bachman  =  cw  *  (1.513  -  8.24  e-4  *  rho  +  3.2249  e-7  *  rho**2)  -  cp
        return
        end

        function  absorp  (rho_w,  rho_b)
C                Author:  Marshall  Hall,  DSTO  Sydney
C          COPYRIGHT      COMMONWEALTH  OF  AUSTRALIA    1992
C  For  sediment  whose  dilational  speed  Cp  is  known,
C  uses  Hamilton's  (1972)  regression  equations  to  get  Cayp
        data  calcite  /  0.3  /     clay  /  0.1  /      ans_sed  /  'U'  /
        quartz  =  1  -  calcite  -  clay
        RHO_g  =  quartz  *  2650  +  calcite  *  2710  +  clay  *  2720
        poros  =  (rho_g  -  rho_b)  /(rho_g  -  rho_w)
C   Attenuation  coefficients  (Hamilton,  1972,  fig.  5):
        if  (poros  .lt.  0.36  .or.  poros  .gt.  0.9)  then
          write  (*,*)  'Warning:  Porosity  of',  nint(100  *  poros),
       :  '  %  is  out  of  bounds  for  Hamilton''s  set  of  regression  equations
       :  for  Cayp'
        end  if
c  To  give  a  continuous  function  for  small  porosity,  Hamilton's  expression  for
c  cayp  at  poros  <  .467,  viz
c                                  CAYP  =  0.2747  +  0.527  *  poros
c   is  replaced  by  (at  poros  =  .36,  it  gives  cayp  =  .31  instead  of  .46):
                                   absorp  =  0.521  *  (poros  /.467)**2
        IF  (poros  .GT.  0.467)  absorp  =  4.903  *  poros  -  1.7688
        IF  (poros  .GT.  0.52)  absorp  =  3.3232  -  4.89  *  poros
        IF  (poros  .GT.  0.65)  absorp  =  0.7602  -  1.487  *  poros  +
     :                                              0.78  *  poros**2
        return
        end
```

```
C      File:  [hall.shallow.stokes]  modes.for


         SUBROUTINE MODES (MAXOUT)      ! Version 4    Aug 92
C                 Author: Marshall Hall, DSTO Sydney
C   Amended by Jonathan Carter to remove the arrays X(SIM) and Y(SIM)
C        COPYRIGHT       COMMONWEALTH OF AUSTRALIA    1992
C   INDICES ARE USED AS FOLLOWS:
C          S -   STEPPING SEARCH FOR NEIGHBOURHOOD OF EIGENVALUE
C         MODE   - MODE NUMBER
C          J   - ITERATIVE SEARCH FOR EXACT EIGENVALUE
C         N   - LAYER NUMBER
         INCLUDE 'PARAM.FOR'
        PARAMETER (SIM = 999)    !  Maximum no. of steps to find neighbourhoods
        REAL*8 CMIN, CMAX
        INTEGER STEP
         character ans_manual
         DIMENSION VEST(5)
         INCLUDE 'COMMON.FOR'
        DATA ITERATE / 40 /       !  Maximum number of iterations to get exact Xi
         namelist / mode_in / ans_manual, maxin, MODE_RITE, TOLERANCE,
     :                          vest, ch_rite
        READ (jin, mode_in)
         call UPPERCASE(ANS_manual)
        IF (maxin .EQ. 0.) maxin = mim       ! default max no. of modes to be
sought
C Tolerance (default maximum error in Xi) is made to decrease as frequency
C increases (see User Guide for explanation)
         IF (TOLERANCE .EQ. 0.) TOLERANCE = 5.E-4 * 32 /freq
        WRITE (*,135) MAXIN
  135 FORMAT (' Calling subroutine MODES; with maximum number of modes
     : specified as', i5)
        WRITE (*, 145) TOLERANCE
  145 FORMAT (' Tolerance in YCF is', 1p e9.1)
       JFAULT = 0          ! flag for problem in subroutine CHAREQ
        DH = DZ(NLAY+1) - DZ(1)
C The step size WDX needs to be reduced as the cut-off frequency is approached.
c Factor in denominator of WDX:
         DIVIDE = MAX (10.,100 * freq_cut /FREQ)
       MAXOUT = MAXIN
        CMIN = MIN (CT(1), CB(1))
       DO 100 N = 2, NWAT
  100 CMIN = MIN (CMIN, CT(N), CB(N))

        CMAX = MAX (CT(1), CB(1))
       DO 120 N = 2, NWAT
  120 CMAX = MAX (CMAX, CT(N), CB(N))
C For the case of an isothermal surface-duct, calculate the maximum wavenumber
C that can be tried without causing overload in the AIRY function subroutine
C (ZETA_MAX = 25.56):
        IF (GEE(1) .LT. 0.) THEN
           r_INIT = MIN(OMEGA /CMIN, SQRT((OMEGA/CMAX)**2 + 25 * GEE(1)**2))
      ELSE
          r_INIT = OMEGA /CMIN
      END IF
        if (dreal(wcs_h) .eq. 0.d0) then
        h_init = 0
      else
c if cs_h > 0, then imag (Xi) < 0 even if the Cayp's are zero, so define:
           h_init = - (min(real(wcs_h /wcp_h), real(wcp_h /wcs_h)) )**3
     :                 * (freq_cut /freq)**3
      end if
        x_init = cmplx(r_init, h_init)
```

```
C     File:  [hall.shallow.stokes]  modes.for

C  An  alternative  ("manual")  method  is  provided  for  finding  modes  that  are  not
C  found  by  the  stepping  search  (this  is  usually  necessary  near  the  cut-off
C  frequency).  The  phase-speed  may  be  estimated  by  extrapolation  from  the
C  phase-speeds  at  higher  frequencies.

       DO 700 MODE = 1, MAXIN
         WDX = - sqrt(real(mode)) /DIVIDE /X_INIT * (PI /DH)**2
        IF (Ans_Manual .EQ. 'Y') THEN
         if (real(vest(mode)) .eq. 0.d0) go to 580
         XSTART = OMEGA /VEST(MODE)
       ELSE

C   step  down  from  X_initial  (= Omega/"Cmin"  if  MODE  =  1,  else  Wig(MODE-1)):
            XSTART = X_INIT + WDX
           CALL CHAREQ (XSTART, YSTART, JFAULT, ch_rite)
        IF (JFAULT .EQ. 1) GO TO 580
         IF (MODE_RITE .EQ. 1) WRITE (JOUT,505) 1, OMEGA /XSTART, YSTART
  505  format (i6, 2f15.6, 5x, 1p 2e15.6)
       XOLD = XSTART
       YOLD = YSTART
           RINDOW = OMEGA * (1./CMIN - DREAL(1./WCP_H))
         DO 270 STEP = 2, SIM
          DISTANCE = OMEGA /CMIN - DREAL(XOLD)
C   If  the  eigenvalue  is  not  close  to  WCP_H:
        IF (DISTANCE .LT. 0.9 * RINDOW) THEN
         XCURRENT = XOLD + WDX
         IF (OMEGA * REAL(1 /XCURRENT) .LE. CMIN) GO TO 580
         IF (REAL(XCURRENT) .LE. 0.) GO TO 580
           CALL CHAREQ (XCURRENT, YCURRENT, JFAULT, ch_rite)
         IF (JFAULT .EQ. 1) GO TO 580
          IF (MODE_RITE .EQ. 1) WRITE (JOUT,505) STEP, OMEGA /XCURRENT,
     :    YCURRENT
C   then,  when  starting  from  XINIT,  the  eigenvalue  is  near  where  REAL(Y)  changes
C   sign  the  second  time:
          IF (REAL (YCURRENT) / REAL (YSTART) .LT. 0.D0) GO TO 260
          IF (REAL (YCURRENT) / REAL (YOLD) .LT. 0.D0) GO TO 300
C  But  if  the  eigenvalue  is  close  to  WCP_H,  then  it  is  near  the  first  sign
change:
       ELSE
         XCURRENT = XOLD + WDX
           CALL CHAREQ (XCURRENT, YCURRENT, JFAULT, ch_rite)
         IF (JFAULT .EQ. 1) GO TO 580
          IF (MODE_RITE .EQ. 1) WRITE (JOUT,505) STEP, OMEGA /XCURRENT,
     :    YCURRENT
          IF (REAL (YCURRENT) / REAL (YOLD) .LT. 0.D0) GO TO 300
       END IF
  260  XOLD = XCURRENT
       YOLD = YCURRENT
  270    CONTINUE
       go to 580
  300  XSTART = XOLD
       END IF

C   CALCULATE  Y  (= L.H.S.  OF  CHARACTERISTIC  EQN)  FOR  2  APPROX  VALUES  OF  XI
        CALL CHAREQ (XSTART, YSTART, JFAULT, ch_rite)
        IF (JFAULT .EQ. 1) GO TO 580
         IF (MODE_RITE .EQ. 1) WRITE (JOUT,505) 1, OMEGA /XSTART, YSTART
        XNEWSTART = XSTART + WDX /DIVIDE
         CALL CHAREQ (XNEWSTART, YNEWSTART, JFAULT, ch_rite)
        IF (JFAULT .EQ. 1) GO TO 580
         IF (MODE_RITE .EQ. 1) WRITE (JOUT,505) 2, OMEGA /XNEWSTART,
     :    YNEWSTART

C   WITH  2  VALUES  OF  Y  CALCULATED,  NOW  iterate  TO  FIND  "EXACT"  VALUES  OF  XI
       XVOLD = XSTART
       YVOLD = YSTART
       XOLD = XNEWSTART
       YOLD = YNEWSTART
```

```
C      File:  [hall.shallow.stokes]  modes.for

        DO 500 J = 3, ITERATE
         IF (YOLD .EQ. YVOLD) GO TO 580
         XCURRENT = (XVOLD * YOLD - XOLD * YVOLD) /(YOLD - YVOLD)
         IF (OMEGA * REAL(1 /XCURRENT) .LE. CMIN) GO TO 580
         IF (REAL(XCURRENT) .LE. 0.) GO TO 580
          CALL CHAREQ (XCURRENT, YCURRENT, JFAULT, ch_rite)
         IF (JFAULT .EQ. 1) GO TO 580
          IF (MODE_RITE .EQ. 1) WRITE (JOUT,505) J, OMEGA /XCURRENT,
     :      YCURRENT
          IF (ABS(YCURRENT) .LT. TOLERANCE) GO TO 550
        XVOLD = XOLD
        YVOLD = YOLD
        XOLD = XCURRENT
        YOLD = YCURRENT
  500  CONTINUE
  580 MAXOUT = MODE - 1
       RETURN
  550 if (mode .ge. 2) then
           if (abs(xcurrent - wig(mode-1)) .le. tolerance) then
            write (*,*) 'Mode ', mode, ' has converged to the same as Mode',
     :     mode - 1
          go to 580
         end if
       end if

         WIG(MODE) = XCURRENT

         IF (MODE_RITE .EQ. 1) WRITE (JOUT,*)
         IF (MODE .LT. MAXIN .AND. MODE_RITE .EQ. 1)
       : WRITE (JOUT,*) ' Mode:', MODE+1
         write (*,*) 'Mode', mode, '   located.'
  700  X_INIT = WIG(MODE)
       RETURN
       END
```

```
c               File: [hall.shallow.stokes] chareq.for


         SUBROUTINE CHAREQ (XI, YCF, jfault, ch_rite)
c    CALCULATES YCF = CHARACTERISTIC FUNCTION FOR GIVEN "XI"
c    and calculates the coefficients A and B in the depth-function expressions
c                  Author: Marshall Hall, DSTO Sydney
c          COPYRIGHT       COMMONWEALTH OF AUSTRALIA    1992
c    Indices: MODE - modes; N - layer

c  !! Note: WPHI is solution to wave equation in a medium in which density (rho)
c  !!  varies. Therefore, Displacement Potential = WPHI /sqrt(rho) /omega^2
c        and Stress = - sqrt(rho) * WPhi.
c      Wphi is called the "Bergmann" potential

        INCLUDE 'PARAM.FOR'
        real*8 real_part
        INCLUDE 'COMMON.FOR'
       VEE = OMEGA /XI

c    WATER COLUMN          "_W"    (density RHO_W assumed constant)

       N = 1
        IF (CB(N) .NE. CT(N)) THEN
        ZT = OMEGA**2 * (1 /VEE**2 - 1 /CT(N)**2) /GEE(N)**2
            CALL AIRY (ZT, XAI , XBI, XAIP, XBIP, 1)
          WA(N,MODE) =    XAI
        if (ch_rite .eq. 1) WRITE (jout,*) 'N:', N,
      :  '   Mode:', MODE,' ABS (WA):', ABS(WA(N,MODE))
          WB(N,MODE) = - XBI
        ZB = OMEGA**2 * (1 /VEE**2 - 1 /CB(N)**2) /GEE(N)**2
            CALL AIRY (ZB, XAI , XBI, XAIP, XBIP, 3)
        THRESH    = LOG(abs(WA(N,MODE))) + LOG(abs(XBI))
         THRESH_P  = LOG(abs(WA(N,MODE))) + LOG(abs(XBIP))
        IF (MAX(THRESH, THRESH_P) .GE. 88.) THEN ! exp(88) is maximum allowed
        JFAULT = 1
         RETURN
          END IF

       WPHI    =               WA(N,MODE) * XBI   + WB(N,MODE) * XAI
        WPHIDZ = - GEE(N)*(WA(N,MODE) * XBIP + WB(N,MODE) * XAIP)

       ELSE

         WB(N,MODE) = (0.D0, 0.D0)
! if mode phase speed V < c(0), the coefficients grow with layer no.; so
!   they are started from a low value
         if (real(vee) .lt. ct(1)) then
           real_part = exp(2 * log(10.) * (real(vee) - ct(1)))
        else
          real_part = 1.d0
        end if
          WA(N,MODE) = dcmplx(real_part , 0.D0)
        WGAM = OMEGA * SQRT(1 /CT(N)**2 - 1 /VEE**2)
        WARG = WGAM * (DZ(N+1) - DZ(N))
             THRESHOLD = abs(dimag(warg))
        IF (THRESHOLD .GE. 88.) THEN
         write (*,*) 'Imag gamma * h > 88. If the number of modes found
        :is unduly small, then examine the sound-speed profile or the
        :geoacoustic model with a view to simplifying either of them'
         JFAULT = 1
          RETURN
           END IF
         WPHI = wa(n,mode) * SIN(WARG)
         WPHIDZ = WGAM * wa(n,mode) * COS(WARG)
       END IF
```

```
      DO 200 N = 2, NWAT
       IF  (CB(N)  .NE.  CT(N))  THEN
        ZT  =  OMEGA**2  *  (1  /VEE**2  -  1  /CT(N)**2)  /GEE(N)**2
c   Values  of  WA  &  WB  in  layer  N  are  prescribed  by  continuity  of  WPHI  &
c   WPHIDZ  at  interface  between  layers  N-1  &  N  (Rho  is  constant  in  water)
         CALL  AIRY  (zt,  XAI  ,  XBI,  XAIP,  XBIP,  3)

         WA(N,MODE)  =  -  (WPHI   *  GEE(N)  *  XAIP
     :                       +  WPHIDZ  *  XAI)  /WRON  /GEE(N)
       if  (ch_rite  .eq.  1)  WRITE  (jout,*)  'N:',  N,
     :  '   Mode:',  MODE,'  ABS  (WA):',  ABS(WA(N,MODE))
       if  (abs(wphidz)  .gt.  0.)
     :     THRESHOLD  =  LOG(abs(wphidz))  +  LOG(abs(XBI))
       IF  (THRESHOLD  .GE.  88.)  THEN
        write  (*,*)  'Phidz  *  Bi  >  exp(88).  If  the  number  of  modes  found
     :is  unduly  small,  then  examine  the  sound-speed  profile  or  the
     :geoacoustic  model  with  a  view  to  simplifying  either  of  them'
        JFAULT  =  1
         RETURN
          END  IF
       WB(N,MODE)  =     (WPHIDZ  *  XBI
     :                    +  WPHI  *  GEE(N)  *  XBIP)  /WRON  /GEE(N)

       ZB  =  OMEGA**2  *  (1  /VEE**2  -  1  /CB(N)**2)  /GEE(N)**2
          CALL  AIRY  (ZB,  XAI  ,  XBI,  XAIP,  XBIP,  3)
       IF  (ABS(WA(N,MODE))  .NE.  0.)
     :                THRESHOLD  =  LOG(abs(WA(N,MODE)))  +  LOG(abs(XBI))
       IF  (THRESHOLD  .GE.  88.)  THEN
        write  (*,*)  'A(n,m)  *  Bi  >  exp(88)'
       JFAULT  =  1
        RETURN
         END  IF

       WPHI    =              WA(N,MODE)  *  XBI   +  WB(N,MODE)  *  XAI
         WPHIDZ  =  -  GEE(N)*(WA(N,MODE)  *XBIP  +  WB(N,MODE)  *XAIP)
      ELSE
        WGAM  =  OMEGA  *  SQRT(1  /CT(N)**2  -  1  /VEE**2)
       WB(N,MODE)  =  WPHI
        WA(N,MODE)  =  WPHIDZ  /WGAM
        if  (ch_rite  .eq.  1)  WRITE  (jout,*)  'N:',  N,
     :  '   Mode:',  MODE,'  ABS  (WA):',  ABS(WA(N,MODE))
        WARG  =  WGAM  *  (DZ(N+1)  -  DZ(N))
        WPHI   =   WA(N,MODE)  *  SIN(WARG)  +  WB(N,MODE)  *  COS(WARG)
        WPHIDZ  =  WGAM  *  (WA(N,MODE)  *  COS(WARG)  -
     :                             WB(N,MODE)  *  SIN(WARG))
      END  IF
  200  CONTINUE
       IF  (ABS(WPHIDZ)  .EQ.  0.)  RETURN

       WGAM_W  =  OMEGA  *  SQRT(1  /CB(NWAT)**2  -  1  /VEE**2)
       WCOST_W  =  SQRT(1  -  (CB(NWAT)  /VEE)**2)
       IF  (WCOST_W  .EQ.  (0.,  0.))  THEN
        WREFLECT  =  (-1.,  0.)
       GO  TO  900
      END  IF

       Z_W  =  RHO_W  *  CB(NWAT)  /WCOST_W            !   Impedance  in  water  ("W")
c - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

```
C                File:  [hall.shallow.stokes]  chareq.for

C    LAYER  (liquid) "_L"  density  RHO_L
        IF  (THICK_L  .GT.  0.)  THEN
C    In  the  (homogeneous)  layer,  the  depth  function  is  expressed  as
C      WPHI  = A * exp [nu  *  (z - z_L)] + B * exp [- nu * (z - z_L)]
C    ( 0 < phase(nu) < 3 deg.  for  V << Cp)
           WNU_L  =  OMEGA  *  SQRT(1 /VEE**2 - 1 /WCP_L**2)
            if (ch_rite .eq. 1) WRITE (jout,*) 'Phase(nu_L):',
     :                RA_TO_D  *  atan2(dimag(wnu_L), dreal(wnu_L))
           WCOSTSQ_L  =  1  -  (WCP_L  /VEE)**2
           WCOST_L  =  SQRT  (WCOSTSQ_L)           !  cos  (theta_L)
           Z_L  =  RHO_L  *  WCP_L  /WCOST_L       !  Impedance  liquid  layer  ("L")
        END IF

c - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
C    HALF-SPACE  (solid  or  liquid)            density  RHO_H
c   compr.  wave's  cos  (theta_h):
        WCOSTSQ_H  =  1  -  (WCP_H  /VEE)**2
         WCOST_H(mode)  =  SQRT  (WCOSTSQ_H)
c  to make the case of a real SSP the limit as Imag part approaches zero for a
c  complex  SSP,  the  following  condition  needs  to  be  inserted
c  [it  converts  a  growing  Z  (imag  gamma  >  0)  into  an  upcoming  Z  (real
c  gamma  <  0).  That  is,  if  phase  gamma^2  le  180  deg,  make  phase  gamma  <  0]
           if  (dimag(wcostsq_h)  .ge.  0.d0  .and.  real(wcostsq_h)  .lt.  0.d0)
     :       WCOST_H(mode)  =  -  SQRT  (WCOSTSQ_H)

c    shear  wave's  cos  (g_h):
        WCOSGSQ_H  =  1  -  (WCS_H  /VEE)**2
         WCOSG_H(mode)  =  SQRT  (WCOSGSQ_H)
c  if  phase  g_h^2  le  180  deg,  make  phase  g_h  <  0  [Zs(z)  to  decay]
          if  (dimag(wcosgsq_h)  .ge.  0.d0  .and.  real(wcosgsq_h)  .lt.  0.d0)
     :     WCOSG_H(mode)  =  -  SQRT  (WCOSGSQ_H)

        XZP  =  RHO_H  *  WCP_H  /WCOST_H(mode)
        XZS  =  RHO_H  *  WCS_H  /WCOSG_H(mode)
       XSING  =  WCS_H  /VEE
       XCOS2G  =  1.  -  2  *  XSING**2
        XSIN2G  =  2  *  XSING  *  WCOSG_H(mode)
        Z_H  =  XZP  *  XCOS2G**2  +  XZS  *  XSIN2G**2    !   Impedance  in  half  space  ("H")
c - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
C    REFLECTIVITY  COEFFICIENTS

        IF  (THICK_L  .GT.  0.)  THEN
         WREF_WL  =  (Z_L  -  Z_W)  /  (Z_L  +  Z_W)

       WREF_LH  =  (Z_H  -  Z_L)  /  (Z_H  +  Z_L)
        WREFLECT  =  (WREF_WL  +  WREF_LH  *  EXP(-  2  *  WNU_L  *  THICK_L))
     :       /  (1.  +  WREF_WL  *  WREF_LH  *  EXP(-  2  *  WNU_L  *  THICK_L))

          ELSE
          WREFLECT  =  (Z_H  -  Z_W)  /  (Z_H  +  Z_W)
        END  IF
         if  (ch_rite  .eq.  1)  WRITE  (jout,*)  'VEE:',  REAL  (VEE),
     :   'WREFLECT:',   WREFLECT

C    The  Characteristic  Function  is  expressed  in  terms  of  WPHI  and  WPHIDZ  at  the
C    bottom  of  the  water  column;  and  the  reflectivity  of  that  interface

  900  YCF  =   WPHI  /WPHIDZ  +  (1  +  WREFLECT)  /WI  /WGAM_W  /
     :                                   (1  -  WREFLECT)
```

```
C               File: [hall.shallow.stokes] chareq.for

Calculation of coefficients WA and WB in the seabed
C   These coefficients are needed in subroutine NORMGROUP; and in FNCTNS if
C   either of the sensors is below the seafloor

C   Layer
C   A and B are obtained from continuity of:
C     vertical particle displ.  [WPHIDZ /sqrt(rho)];  and
C     stress (- w^2 * sqrt(rho) * WPHI) at z = z_L:
         if (thick_l .gt. 0.) then
         N = NWAT + 1
          if (ch_rite .eq.1) write (jout,*) WPHI * SQRT(RHO_W/RHO_L)/2
          if (ch_rite .eq.1) write (jout,*) WPHIDZ /SQRT(RHO_W/RHO_L)
     :                                                      /WNU_L /2
            WA(N,MODE) = WPHI * SQRT(RHO_W /RHO_L) /2 +
      :                    WPHIDZ /SQRT(RHO_W /RHO_L) /WNU_L /2
         THRESH    = LOG(abs(WA(N,MODE))) + WNU_L * THICK_L
         IF (THRESH .GE. 88.) RETURN
         if (ch_rite .eq. 1) WRITE (jout,*) 'N:', N,
       : '   Mode:', MODE,' ABS (WA):', ABS(WA(N,MODE))
         if (ch_rite .eq. 1) WRITE (jout,*)
             WB(N,MODE) =    WPHI * SQRT(RHO_W /RHO_L) - WA(N,MODE)
         THRESH    = LOG(abs(WB(N,MODE))) - WNU_L * THICK_L
         IF (THRESH .GE. 88.) RETURN
              WPHI_L = WA(N,MODE) * exp (WNU_L * THICK_L) +
      :                    WB(N,MODE) * exp (- WNU_L * THICK_L)
             IF (DREAL(VEE) .GT. DREAL(WCP_L)) THEN
               WPHIDZ_L = WNU_L * (WA(N,MODE) * exp (WNU_L * THICK_L) -
      :                    WB(N,MODE) * exp (- WNU_L * THICK_L))
              YCF =  WPHI_L /WPHIDZ_L + (1 + WREF_LH) /WNU_L /(1 - WREF_LH)
             END IF

              end if

C   half-space
        N = NLAY + 1
         WA(N,MODE) = (0.D0, 0.D0)
         if (thick_l .GT. 0.) then
          WB(N,MODE) =   WPHI_L * SQRT(RHO_L /RHO_H)
          else
          WB(N,MODE) =   WPHI * SQRT(RHO_W /RHO_H)
        end if

        RETURN
        END
```

```
C     File:  [hall.shallow.stokes]  normgrou.for


         SUBROUTINE  NORMGROUP(ANS_GROUP,  NORM_RITE)
C                 Author: Marshall Hall, DSTO Sydney
C         COPYRIGHT      COMMONWEALTH OF AUSTRALIA    1992

C   INDICES  ARE  USED  AS  FOLLOWS:
C          MODE   - MODE NUMBER
C          N   - LAYER NUMBER
         INCLUDE  'PARAM.FOR'
         include  'common.for'
         if  (norm_rite  .eq.  1)  WRITE  (JOUT,105)  MODE
  105  FORMAT  (/'   Mode:',  i3/)
        WN(MODE)  =  (0.D0,  0.D0)
        VEE  =  OMEGA  /WIG(MODE)
        XI  =  WIG(MODE)
C   contributions  of  the  NWAT  water  layers:
       DO 100 N = 1,  NWAT
         IF  (NORM_RITE  .EQ.  1)  WRITE  (JOUT,115)  N,   WA(N,MODE),  WB(N,MODE)
  115   FORMAT  ('  Layer:',  i4,   '   WA:',  1p 2e10.1,'   WB:',  2e10.1)

        IF  (CB(N)  .NE.  CT(N))  THEN
          ZT  =  OMEGA**2  *  (1  /VEE**2  -  1  /CT(N)**2)  /GEE(N)**2
          CALL AIRY (ZT,  XAI ,  XBI,  XAIP,  XBIP,  3)
          WPHI   =  WA(N,MODE)  *  XBI   +  WB(N,MODE)  *  XAI
          WPHIP  =  WA(N,MODE)  *  XBIP  +  WB(N,MODE)  *  XAIP
          WDELN(n)  =  -  (-  ZT  *  WPHI**2  +  WPHIP**2)  /  GEE(N)

          ZB  =  ZT  -  GEE(N)  *  (DZ(N+1)  -  DZ(N))
          CALL AIRY (ZB,  XAI ,  XBI,  XAIP,  XBIP,  3)
          WPHI   =  WA(N,MODE)  *  XBI   +  WB(N,MODE)  *  XAI
          WPHIP  =  WA(N,MODE)  *  XBIP  +  WB(N,MODE)  *  XAIP
          temp1  =  real(log(-  ZB))  +
     :              2  *  real(log(WPHI))  -  log(abs(gee(n)))
           temp2  =  2  *  real(log(wphip))  -  log(abs(gee(n)))
           if  (max(temp1,temp2)  .lt.  86.)  then
            WDELN(n)  =  WDELN(n)  +  (-  ZB  *  WPHI**2  +  WPHIP**2)  /  GEE(N)
        else              !  The  "average"  value  of  the  depth  function  in
          wdeln(n)  =  0.          !  such  a  layer  will  be  negligible
        end if
          WN(MODE)  =  WN(MODE)  +  WDELN(n)

       ELSE

          WGAM  =  OMEGA  *  SQRT(1/CT(N)**2  -  1/VEE**2)
          THICK  =  DBLE(DZ(N+1)  -  DZ(N))
         WARG  =  WGAM  *  THICK
          temp1  =  2  *  log(abs(wa(n,mode)))
           if  (abs(wb(n,mode))  .ne.  0.)  temp2  =  2  *  log(abs(wb(n,mode)))
          temp3  =  dimag(2  *  warg)
           if  (max(temp1,  temp2,  temp3,  temp1+temp3,  temp2+temp3)  .lt.  87.)
     :  then
            WDELN(n)  =  (WA(N,MODE)**2  +  WB(N,MODE)**2)  *  THICK  /2
     :           +  (WB(N,MODE)**2  -  WA(N,MODE)**2)  *  SIN  (2  *  WARG)  /4  /WGAM
     :            +  WA(N,MODE)  *  WB(N,MODE)  /2  /WGAM  *  (1  -  COS(2  *  WARG))
        else
          wdeln(n)  =  0
        end if
          WN(MODE)  =  WN(MODE)  +  WDELN(n)
       END  IF

           if  (norm_rite  .eq.  1)  WRITE  (JOUT,125)  WDELN(n)
  125   FORMAT  (13x,  '  Del  Norm:',  1p  2e9.1/)

  100  CONTINUE
```

```
c     File:  [hall.shallow.stokes]  normgrou.for

c   Contributions  of  the  (liquid)  layer:
        IF  (THICK_L  .GT.  0.)  THEN
         N = NWAT + 1
         IF  (NORM_RITE  .EQ.  1)  WRITE  (JOUT,115)  N,   WA(N,MODE),  WB(N,MODE)
           WNU_L = OMEGA * SQRT(1 /VEE**2 - 1 /WCP_L**2)
          WARG = WNU_L * THICK_L
          temp1 = 2 * log(abs(wa(n,mode)))
           if  (abs(wb(n,mode))  .ne.  0.)  temp2 = 2 * log(abs(wb(n,mode)))
         temp3 = dimag(2 * warg)

           if  (max(temp1,  temp2,  temp3,  temp1+temp3,  temp2+temp3)  .ge.  87.)
     :    then
          WRITE  (*,705)  NINT(THICK_L),  NINT(FREQ)
 705  FORMAT  ('  For  this  case  (Layer  thickness  =',  i3,
     :           '   and  Frequency  =',  i4,  ',  numerical  limitations  cause  the
     :amplitude  of  the  depth  function  at  the  bottom  of  the  layer  to  be
     :sufficiently  high  as  to  result  in  numerical  overflow.')
          WRITE  (*,*)  '  For  this  frequency,  the  boundary  between  the  layer
     :  and  the  half-space  should  have  a  negligible  effect
     :  on  the  seafloor  reflectivity'
         WRITE  (*,*)  '"Remove"  the  layer  by  (1)  assigning
     :  its  properties  to  the  Half-space;  and'
         WRITE  (*,*)  '(2)  setting  THICK_L = 0'
         WRITE  (*,*)
         wdeln(n) = 0

      else

           WDELN(n) = 2 * WA(N,MODE) * WB(N,MODE) * THICK_L
     :           + WA(N,MODE)**2 * (EXP(  2 * WARG) - 1) /2 /WNU_L
     :           - WB(N,MODE)**2 * (EXP(- 2 * WARG) - 1) /2 /WNU_L
      end if
        IF  (NORM_RITE  .EQ.  1)  WRITE  (JOUT,125)  WDELN(n)
          WN(MODE) = WN(MODE) + WDELN(n)

      END IF            !  end  of  the  "if  Thick_L  >  0"  detour
c - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

```
C      File:  [hall.shallow.stokes]  normgrou.for

C   Contribution  of  the  half-space.   gam  =  k  cos  theta,  Nu  =  i  *  gam.
          WGAM_H  =  OMEGA  /WCP_H  *  wcost_h(mode)
           write  (*,205)  MODE,  RA_TO_D  *  atan2(dimag(wGAM_h),  dreal(wGAM_h))
   205  format  ('  Mode:',  i5,  '  Phase  Gam_h  (deg.):',  f9.4)
         WNU_h  =  WI  *  WGAM_H
          wnusq_h  =  wnu_h**2
         n  =  nlay  +  1
            IF  (NORM_RITE  .EQ.  1)  WRITE  (JOUT,115)  N,   WA(N,MODE),  WB(N,MODE)
            if  (abs(wa(n,mode))  .ne.  0.)  templ  =  2  *  log(abs(wa(n,mode)))
           temp2  =  2  *  log(abs(wb(n,mode)))

           if  (max(templ,  temp2)  .lt.  87.)  then
C    If  half-space  is  liquid,  use  simple  formulas  for  normalization:
           IF  (DREAL(WCS_H)  .EQ.  0.D0)  THEN
            WDELN(n)  =  wb(n,MODE)**2  /2  /WNU_h
          ELSE
C   If  half-space  is  solid  then  Normalization  is  computed  from  Eq.  (6)  in  Ellis
c   &  Chapman  (1985),  adapted  to  a  time  dependence  of  exp(+iwt)
c   (See  also  Eq.  (E53)  in  my  "book")
C   Sig  is  the  the  shear  counterpart  to  Nu:  Sig  =  i  *  Gam_S,  Gam_s  =  ks  cos  g

            WGAM_S  =  OMEGA  /WCS_H  *  wcosg_h(mode)
             write  (*,215)  RA_TO_D  *  atan2(dimag(wGAM_S),  dreal(wGAM_S))
   215     format  ('  Phase  GAM_S  (deg.):',  f8.4)
           WSIG  =  WI  *  WGAM_S
           wsigsq  =  wsig**2
           xisq  =  xi**2
            w_thetap  =  1  -  2  *  xisq  /(xisq  -  wsigsq)  *  (1  -  2  *  wnu_h  *  wsig
     :        /(xisq  +  wsigsq))                        !  my  Eq.  (E43)
            w_eta  =  1./  2  /WNU_h  -  4  *  wnu_h  /(xisq  +  wsigsq)    !  my  Eq.  (E38)
     :         +  2  *  wnusq_h  /wsig  *  (xisq  +  2  *  wsigsq)  /(xisq  +  wsigsq)**2
            WDELN(n)  =  w_eta  *  wb(n,mode)**2  /  w_thetap**2           !  my  Eq.  (E53)
          END  IF

         else
          wdeln(n)  =  0
         end  if

           IF  (NORM_RITE  .EQ.  1)  WRITE  (JOUT,125)  WDELN(n)

          WN(MODE)  =  WN(MODE)  +  WDELN(n)
c   note  that  n  =  nlay  +  1  throu-out  the  half-space  section.
           IF  (ANS_GROUP  .EQ.  'Y')  call  group_speed(w_thetap)

         RETURN
         END
```

```
C      File:  [hall.shallow.stokes]  normgrou.for

           SUBROUTINE  GROUP_speed(w_thetap)
C                 Author: Marshall Hall, DSTO Sydney
C          COPYRIGHT      COMMONWEALTH OF AUSTRALIA    1992
C     INDICES ARE USED AS FOLLOWS:
C          MODE   - MODE NUMBER
C          N   - LAYER NUMBER
          include 'param.for'
C     Variables for the NAG integration routine D01AJF:
         PARAMETER (LW = 800, LIW = LW /8 + 2)
          REAL*8 RFUNCT, HFUNCT, THICK, EPSABS, EPSREL, ABSERR, INTEGRAL,
     :         WORK(LW)
         INTEGER IW(LIW)
          COMPLEX*16 DENOM, DENOM_LIQ, DENOM_SHEAR
          EXTERNAL RFUNCT, HFUNCT, D01AJF
         COMMON / TELNUM / KOUNT
          INCLUDE 'COMMON.FOR'

         DATA  EPSREL / 1. D-3 / EPSABS / 0.D0 /
         DENOM = (0.D0, 0.D0)
C  Calculation of the mode group speed:
C  The following statements include the NAG (real*8) function D01AJF to calculate
C  the integral of external function FUNCT from 0. to THICK to a relative
C  accuracy of EPSR. The parameters NEVAL and RELERR are determined by the
C  function. IFAIL is re-valued by the function if there is a failure.
C  D01AJF is required if the mode Group Speed is needed.
C  The source code for RFUNCT (real part) and HFUNCT (imag part) are located
C  after the end of this subroutine.
C   If an integration routine is unavailable, then the call to one should be
C  deleted
         do 100 n = 1, nwat
          if (cb(n) .ne. ct(n)) then
           THICK = DBLE(DZ(N+1) - DZ(N))
           DENOM = DENOM + WDELN(n) /CT(N)**2
          KOUNT = 0
          IFAIL = 1
           DEE = ((CT(N) /CB(N))**2 - 1) /THICK
            CALL D01AJF(RFUNCT, 0.D0, THICK, EPSABS, EPSREL, RINTEGRAL,
     :        ABSERR, WORK, LW, IW, LIW, IFAIL)
            CALL D01AJF(HFUNCT, 0.D0, THICK, EPSABS, EPSREL, HINTEGRAL,
     :        ABSERR, WORK, LW, IW, LIW, IFAIL)
            DENOM = DENOM + DCMPLX(RINTEGRAL, HINTEGRAL) * DEE /CT(N)**2
         else
           DENOM = DENOM + WDELN(n) /CT(N)**2
         end if
   100 continue
C  contribution of the (liquid) layer
          if (thick_l .gt. 0.) then
          n = nwat + 1
           DENOM = DENOM + WDELN(n) /WCP_L**2
         end if

C  contribution of the half-space:
         n = nlay + 1
          VEE = OMEGA /WIG(MODE)
          if (dreal(wcs_h) .eq. 0.d0) then          !   liquid half-space
           DENOM = DENOM + WDELN(n) /WCP_H**2
            GROUP(MODE) = REAL(WN(MODE) /VEE /DENOM)
         else                                 !   solid half-space
C   If half-space is solid DENOM_SHEAR is computed from algorithm based
c   on Eq. (9) of Koch et al (1983).
          WKS = OMEGA /WCS_H
          WKP = OMEGA /WCP_H
           WNU_h = WI * OMEGA /WCP_H * wcost_h(mode)
c            Z is depth factor in compressional displacement potential
c           wpee is initial amplitude of Z (at z = H)
c "Denom_liq" = kp^2 * Integral[H,inf] Z^2 dz, where Z = wpee * exp[-nu * (z-H)]
           wpee = wb(n,mode) /sqrt(rho_h) /w_thetap             ! my Eq. (E54)
          DENOM_LIQ = WKP**2 * wpee**2 /2 /WNU_h
```

```
C      File:  [hall.shallow.stokes]  normgrou.for

          xi  =  wig(mode)
          xisq  =  xi**2
           WSIG  =  WI  *  OMEGA  /WCS_H  *  wcosg_h(mode)
          wsigsq  =  wsig**2
           XOE_CSQ  =  2  *  XIsq**2  -  2  *  (XI  *  WKS)**2  +  WKS**4  /4
           XOE_BC  =  WNU_h  *  (2  *  XIsq  -  WKS**2)  +  WSIG  *  (2  *  XIsq  -  WKP**2)
           wce  =  wpee  *  2  *  wnu_h  /(xisq  +  wsigsq)
c  "denom_shear"  =  (cs/vee)^2  *  Integral[H,inf]  P^2  +  2  Q  *  (P'  +  ks^2  V')
c  note:  vee  is  phase-speed,  V  is  depth  factor  in  shear  displacement  potential
c          Q  =  Z  +  V',  and  P  =  2  Q'  +  ks^2  *  V
           DENOM_SHEAR  =  4  *  (WCS_H  /VEE)**2  *  (WNU_h  *  wpee**2
      :          +  XOE_CSQ  *  WCE**2  /2  /WSIG
      :          -  XOE_BC  *  wpee  *  WCE  /(WNU_h  +  WSIG))
C   In  Koch's  Eq.(9),  the  denominator  has  been  multiplied  by  Kp^2  =
C   (omega  /Cp)^2  rather  than  by  1  /Cp^2.  We  therefore  now  divide  the  final-layer
C   component  of  DENOM  by  omega^2:
            DENOM  =  DENOM  +  (DENOM_LIQ  +  DENOM_SHEAR)  /OMEGA**2
            GROUP(MODE)  =  REAL(WN(MODE)  /VEE  /DENOM)
        end  if

         IF  (GROUP(MODE)  .LT.  0.)  THEN
          WRITE  (*,125)  MODE
 125      FORMAT  ('  Group  speed  for  mode',  i3,'  is  negative.')
           WRITE  (*,*)  'This  is  caused  by  computer  numerical  limitations.'
      :          ,'The  sound-speed  profile  needs  to  be  simplified.'
          WRITE  (*,*)  'FFI  see  the  user  manual'
         STOP
        END  IF

        return
        end



         REAL*8  FUNCTION  RFUNCT(ZED)
         INCLUDE  'PARAM.FOR'
        COMMON  /  TELNUM  /  KOUNT
         INCLUDE  'COMMON.FOR'
        KOUNT  =  KOUNT  +  1
         ZETA  =  (WIG(MODE)**2  -  (OMEGA  /CT(N))**2)/GEE(N)**2  -  GEE(N)  *  ZED
         CALL  AIRY  (ZETA,  XAI  ,  XBI,  XAIP,  XBIP,  1)
        WPHI   =  WA(N,MODE)  *  XBI   +  WB(N,MODE)  *  XAI
         RFUNCT  =  DREAL(WPHI**2)  *  ZED
        RETURN
        END



         REAL*8  FUNCTION  HFUNCT(ZED)
         INCLUDE  'PARAM.FOR'
        COMMON  /  TELNUM  /  KOUNT
         INCLUDE  'COMMON.FOR'
        KOUNT  =  KOUNT  +  1
         ZETA  =  (WIG(MODE)**2  -  (OMEGA  /CT(N))**2)/GEE(N)**2  -  GEE(N)  *  ZED
         CALL  AIRY  (ZETA,  XAI  ,  XBI,  XAIP,  XBIP,  1)
        WPHI   =  WA(N,MODE)  *  XBI   +  WB(N,MODE)  *  XAI
         HFUNCT  =  DIMAG(WPHI**2)  *  ZED
        RETURN
        END
```

```
c     File:  [hall.shallow.stokes]   fnctns.for


          SUBROUTINE FNCTNS (maxout, SOURCE, NTRANS, Ans_MODE_FN)
CALCULATES DEPTH FUNCTIONS AT SOURCE AND "NTRANS" RECEIVERS
c   FOR EACH OF "maxout" MODES.
c                Author: Marshall Hall, DSTO Sydney
c        COPYRIGHT     COMMONWEALTH OF AUSTRALIA    1992
c
c   INDICES ARE USED AS FOLLOWS:
c         Mod - MODE NUMBER          N - LAYER NUMBER
c          J - RECEIVER NUMBER
c        js - layer no. of source;   JR(J) - LAYER NUMBER OF THE J'TH RECEIVER
         INCLUDE 'PARAM.FOR'
          dimension RFNCTN(JIM,MIM), HFNCTN(JIM, MIM)
         DIMENSION JR(JIM)
         INCLUDE 'COMMON.FOR'
        data jbin / 72 /
         open (unit = jbin, file = '[.plot]modefns.dat',
     :     status = 'unknown', form = 'unformatted')
         write (jbin) fname
         write (*,*) fname

         write (jbin) ntrans
         write (jbin) maxout

         WRITE (*,115) maxout
  115 FORMAT (' Number of modes:', i4)
c     SOURCE
        DO 420 N = 2, NLAY+1
         IF (SOURCE .LT. DZ(N)) GO TO 430    !determine which layer contains source
  420 CONTINUE
  430 JS = N - 1

c     Source in water column:
        IF (JS .LE. NWAT)  THEN
         DO Mod = 1, maxout
           CALL POTENTIAL (Mod, JS, SOURCE, US(Mod))
         END DO
        END IF
c     Source in Layer:
         IF (THICK_L .GT. 0. .AND. JS .EQ. NWAT+1) THEN
         DO Mod = 1, maxout
          VEE = OMEGA /WIG(Mod)
           WNU = OMEGA * SQRT(1 /VEE**2 - 1 /WCP_L**2)
          WARG = WNU * (SOURCE - DZ(JS))
           US(Mod) = WA(JS,Mod) * EXP(WARG) + WB(JS,Mod) * EXP(- WARG)
         END DO
        END IF
c     Source in Half-space:
         IF (JS .EQ. NLAY+1) THEN
         DO Mod = 1, maxout
          VEE = OMEGA /WIG(Mod)
           WNU = WI * OMEGA /WCP_H * wcost_h(mod)
          WARG = WNU * (SOURCE - DZ(JS))
           US(Mod) = WB(JS,Mod) * EXP(- WARG)
         END DO
        END IF

c    RECEIVERS
d          write (*,*) 'nlay:', nlay
        DO 460 J = 1, NTRANS
          DO 440 N = 2, NLAY+1
d           write (*,*) 'n:', n, '   dz:', dz(n)
            IF (DT(J) .LT. DZ(N)) GO TO 450   ! determine layer that contains
receiver
  440    CONTINUE
  450 JR(J) = N - 1
d        write (*, *) 'j:', j, '   depth:', dt(j), '   jr(j):', jr(j)
  460 CONTINUE
```

```
C      File:  [hall.shallow.stokes]  fnctns.for

        WRITE  (JOUT,445)
  445  FORMAT  (  40('  -')/)
         IF  (Ans_MODE_FN  .EQ.  'P')   WRITE  (JOUT,435)
   435  FORMAT  ('  Absolute  values  of  normalized  depth-functions'/)

        DO  480  J  =  1,  NTRANS
        JV  =  JR(J)
C     Receiver  in  water  column:
        IF  (JV  .LE.  NWAT)   THEN
         DO  Mod  =  1,  maxout
            CALL  POTENTIAL  (Mod,  JV,  DT(J),  UR(J,Mod))
        END  DO
       END  IF
C     Receiver  in  Layer:
         IF  (THICK_L  .GT.  0.  .AND.  JV  .EQ.  NWAT+1)  THEN
         DO  Mod  =  1,  maxout
           VEE  =  OMEGA  /WIG(Mod)
           WNU  =  OMEGA  *  SQRT(1  /VEE**2  -  1  /WCP_L**2)
           WARG  =  WNU  *  (DT(J)  -  DZ(JV))
            UR(J,Mod)  =  WA(JV,Mod)  *  EXP(WARG)  +  WB(JV,Mod)  *  EXP(-  WARG)
        END  DO
       END  IF
C     Receiver  in  Half-space:
        IF  (JV  .EQ.  NLAY+1)  THEN
         DO  Mod  =  1,  maxout
           VEE  =  OMEGA  /WIG(Mod)
           WNU  =  WI  *  OMEGA  /WCP_H  *  wcost_h(mod)
           WARG  =  WNU  *  (DT(J)  -  DZ(JV))
            UR(J,Mod)  =  WB(JV,Mod)  *  EXP(-  WARG)
        END  DO
       END  IF

        DO  470  Mod  =  1,  maxout
          if  (abs(wn(mod))  .eq.  0.d0)  go  to  470
           RFNCTN(J,Mod)  =  DREAL(UR(J,  Mod)  /SQRT(WN(Mod)))
           HFNCTN(J,Mod)  =  DIMAG(UR(J,  Mod)  /SQRT(WN(Mod)))
  470  continue
  480  CONTINUE

       do  j  =  1,  ntrans
          write  (jbin)  dt(j),  (rfnctn(j,mod),  mod  =  1,  maxout),
     :                            (hfnctn(j,mod),  mod  =  1,  maxout)
      end  do

        IF  (Ans_MODE_FN  .EQ.  'P')  THEN
          WRITE  (JOUT,465)  SOURCE,  (DT(J),  J  =  1,  NTRANS)
  465    FORMAT  (20X,  'Source  &  Receiver  Depths  (m)'//  F8.1,  10F9.1)
         WRITE  (JOUT,*)
D         WRITE  (JOUT,475)  JS,  (JR(J),  J  =  1,  NTRANS)
  475   FORMAT  (9i9)
          DO  600  Mod  =  1,  maxout
             WRITE  (JOUT,595)   ABS(US(Mod)  /SQRT(WN(Mod))),
     :                         (ABS(UR(J,  Mod)  /SQRT(WN(Mod))),  J  =  1,  NTRANS)
  595      FORMAT  (1P  10E9.1)
  600    .  CONTINUE
       END  IF

       RETURN
       END
```

```
C       File:   [hall.shallow.stokes]   fnctns.for


        SUBROUTINE  POTENTIAL  (Mod,  LAYER,  DEPTH,  WATERPHI)
       INCLUDE  'PARAM.FOR'
       INCLUDE  'COMMON.FOR'
       VEE  =  OMEGA  /WIG(Mod)
       IF  (CB(LAYER)  .NE.  CT(LAYER))  THEN
         ZETA  =  OMEGA**2  *  (1  /VEE**2  -  1  /CT(LAYER)**2)  /  GEE(LAYER)**2
     :             -  GEE(LAYER)  *  (DEPTH  -  DZ(LAYER))
         CALL  AIRY  (ZETA,  XAI  ,  XBI,  XAIP,  XBIP,  1)
           WATERPHI  =  WA(LAYER,Mod)  *  XBI  +  WB(LAYER,Mod)  *  XAI
          ELSE
           WGAM  =  OMEGA  *  SQRT(1  /CT(LAYER)**2  -  1  /VEE**2)
         WARG  =  WGAM  *  (DEPTH  -  DZ(LAYER))
         WATERPHI  =  WA(LAYER,Mod)  *  SIN(WARG)  +  WB(LAYER,Mod)  *  COS(WARG)
      END  IF
      RETURN
      END
```

```
C     File:  [hall.shallow.stokes]  relint.for


          SUBROUTINE  RELINT(MAXOUT,SOURCE,NTRANS,RMAX,DELR,ans_bli,Ans_RI)
C   CALCULATES Transmission Loss AT EACH OF "NTRANS" Transducer DEPTHS
C   DT(J) and at each range from DELR to RMAX in steps of DELR.
C                Author: Marshall Hall, DSTO Sydney
C        COPYRIGHT     COMMONWEALTH OF AUSTRALIA    1993

C   INDICES ARE USED AS FOLLOWS:
C        M - MODE NUMBER          J - RECEIVER NUMBER
C        N - LAYER NUMBER         K - RANGE NUMBER

        INCLUDE 'PARAM.FOR'
       PARAMETER (KIM = 480)
       COMPLEX*16 S15DDF
        DIMENSION UMP(MIM),    RK(KIM),  tloss(KIM,JIM)
        INCLUDE 'COMMON.FOR'
       data jput / 75 /
C ABSORPTION: Magnesium Sulphate (M); Boric Acid (B); & Magnesium Carbonate (C)
          FK = FREQ /1000.
C   Relaxation frequencies in kHz (Mellen, Scheifele & Browning, 1987):
          FRM = 50   * 10.**(TEMP /60)
          FRB = 0.9  * 10.**(TEMP /70)
          FRC = 4.5  * 10.**(TEMP /30)
C   Contributions of the 3 components (dB /Km):
       AM = 0.5 * FRM * FK**2 /(FRM**2 + FK**2)
       AB = 0.1   * 1.9 * FRB * FK**2 /(FRB**2 + FK**2)
       AC = 0.03 * 1.9 * FRC * FK**2 /(FRC**2 + FK**2)
       ABSORP = AM + AB + AC
        WRITE (JOUT,205)   NINT(FREQ), NINT(TEMP), ABSORP
  205  FORMAT (' Estimated rate of absorption at ',I6,   ' Hz ,for tempera
      :ture of ', I3, ' deg. C'/' is', 1P E8.1, ' dB/Km'/)

       DO 200 M = 1, MAXOUT
        if (abs(wn(m)) .eq. 0.d0) go to 200
        UMP(M) = US(M) / WN(M) / SQRT(WIG(M))
  200  continue
       nrange = NINT(RMAX  /DELR)
       IF (nrange .GT. KIM) nrange = KIM
       RMAX = nrange * DELR
C Calculate terms needed for the Branch Line Integral as per Brekhovskikh
C (1980, pp 332 - 336)
       ratio = rho_h /rho_w
c       WNUSQ = 1 - (CB(NWAT) /WCP_h)**2
c       WKNU = OMEGA /CB(NWAT) * SQRT(WNUSQ)
C Brek's expression for isospeed water column is adapted to the general case:
       warg = (0.d0, 0.d0)
c nlay is no. of layers, including the liquid sediment layer if present:
      do n = 1, nlay
        if (ct(n) .eq. cb(n)) then
c        WARG = WKNU * Delta-H
          warg = warg + OMEGA /CB(N) * SQRT(1 - (CB(N) /WCP_h)**2) *
     :                 (dz(n+1) - dz(n))
       else
c  warg is integral of sqrt(k^2 - kp^2) where k^2(z) = k^2(z0) + g^3 * (z - z0):
         warg=warg + 2 * omega**3 /3 * ((1./cb(n)**2 - 1./wcp_h**2)**1.5
      :            - (1./ct(n)**2 - 1./wcp_h**2)**1.5) /gee(n)**3
       end if
      end do
       wknu = warg /dz(nlay + 1)
c        WKNU = OMEGA /CB(nwat) * SQRT(WNUSQ)
        wnusq = (wknu * cb(nwat) /omega)**2
        WSQRATE = OMEGA /CB(NWAT)**2 * WCP_h /2 * WNUSQ * RATIO**2
      :        * COS(WARG)**2 / SIN(WARG)               ! Eq. (37.27)
      :        /(SIN(WARG) - COS(WARG) * WARG * RATIO**2)
```

```
C      File:  [hall.shallow.stokes]  relint.for

        DO 500 K = 1, nrange
        RK(K) = K * DELR
        RANGE = 1000 * RK(K)
        REF = SQRT(2 * PI /RANGE)

        DO 400 J = 1, NTRANS

        USUM = (0.D0, 0.D0)

C   USUM = Sum ( M=1 TO MAXOUT) US(M) * UR(J, M) * EXP(-i * E(M)*RANGE(K)) /
C                                     D(M) * (E(M))**0.5
        DO 300 M = 1, MAXOUT
         USUM = USUM + UMP(M) * UR(J, M) * EXP(-WI * WIG(M) * RANGE)
  300   CONTINUE
        WESIDUE = - WI * SQRT(WI) * REF * USUM

        XBLI = (0.D0, 0.D0)
        IF (ans_bli .ne. 'N') THEN
        WSQ = WSQRATE * RANGE
C Asymptotic expression for K when WSQ is large, Brekhovskikh Eq.(37.30):
C (Prevents overflow in exp(-x^2) within Erfc function when Imag(x) is large)
         IF (ABS(WSQ) .GT. 88.) THEN   ! exp(88) is VAX's largest number
          WKAY = 1 - 15 /4 /WSQ**2 + WI * (1.5 /WSQ - 105 /8 /WSQ**3)
          ELSE
C General expression for K, Brekhovskikh Eq. (37.29):
            WX = SQRT (WSQ) * EXP (WI * PI /4)
D        WRITE (JOUT,*) 'Erfc argument:', wi * WX
       ifail = 0
d        write (jout, *) 'Erfc * exp(-Arg^2):', s15ddf(wi * wx, ifail)
         if (real(wsq) .ge. 0.) then
          WKAY = 2 * WI * WSQ * (1 - WX * SQRT(PI) * S15DDF(WI * WX,IFAIL))
       else
          WKAY = 2 * WI * WSQ * (1 + WX * SQRT(PI) * S15DDF(-WI* WX,IFAIL))
       end if
       END IF
D         WRITE (JOUT, 315) RANGE, WSQ, WKAY
  315  FORMAT (' Range (m):', F8.0, '   W^2:', F8.3, F8.3, '   K(Brek):',
     : 1p 2e12.3)
         WINT = - WKAY /WNUSQ /COS(WARG)**2                ! Eq. (37.28)
         XBLI = 2 * WI * CB(NWAT)**2 /WCP_h /OMEGA /RATIO /RANGE**2 * WINT
C   For the BLI, the depth functions are assumed to be sinusoidal to fit into
C    Brekhovskikh's formulation:
     :          * SIN(WKNU * SOURCE) * SIN(WKNU * DT(J))
     :         * EXP (- WI * OMEGA /WCP_h * RANGE)              ! Eq. (37.25)
          END IF

        XPSI = WESIDUE + XBLI
         tloss(k,j) = - 20 * log10(abs(xpsi)) + absorp * rk(k)

         IF (K * J .EQ. 1) WRITE (JOUT,375) ABS(XBLI), ABS(WESIDUE)
  375  FORMAT (' At R(1) and DT(1), BLI =',1P E9.1, ';   Mode Sum =',E9.1)
     :
  400  CONTINUE

  500  CONTINUE

         WRITE (JOUT,505) NTRANS, nrange
  505  FORMAT (/' No. of receivers:', i4, 24x, 'No. of ranges:', i5)
```

```
C      File:  [hall.shallow.stokes]  relint.for

         IF  (Ans_RI  .eq.  ' ')  then
          WRITE  (JOUT,*)
          WRITE  (JOUT,*)  '                          TRANSMISSION  LOSSES  (dB  re  m^2)'
          WRITE  (JOUT,*)
           WRITE  (JOUT,465)  SOURCE,  (DT(J),  J  =  1,  NTRANS)
   465    FORMAT  (20X,  'Source  &  Receiver  Depths  (m)'/  F8.1,  10F9.1)
          WRITE  (JOUT,*)
           WRITE  (JOUT,*)  '  RANGE  (Km)'
         DO  K  =  1,  nrange
            WRITE  (JOUT,815)  RK(K),  (tloss(K,J),  J  =  1,  NTRANS)
         end  do
   815    FORMAT  (F8.3,  10F9.1)

        else
            open  (unit  =  jput,  file  =  '[.plot]tloss'//fname(13:14)//'.dat',
       :                             status  =  'unknown',  form  =  'unformatted')
          write  (jput)  fname
          write  (jput)  nrange
          write  (jput)  ntrans
         DO  K  =  1,  nrange
            WRITE  (jput)  RK(K),  (tloss(K,J),  J  =  1,  NTRANS)
         end  do
        end  if

        RETURN
        END
```

Annex E

Adjustments to the VAX Version of STOKES so it can run on a DOS PC

1. Memory Restriction

The rule for working out maximum dimensions in MS-FORTRAN programs in PC-DOS is that no array size can exceed 64 Kilo-bytes. Thus, if you have a real array of dimension X(MIM, KIM) you get the result that

$$4 * MIM * KIM < 64 * 1024$$

(real values are 4 bytes).

File [hall.shallow.stokes] param.for contains the following statements:

```
PARAMETER (NIM = 20, MIM = 300, JIM = 101)
IMPLICIT COMPLEX*16 (U - 2)
```

File [hall.shallow.stokes]common.for contains the following statement:

```
COMMON / FUNCT / WA (NIM, MIM), WB (NIM,MIM), US (MIM),
UR (JIM, MIM)
```

PARAM.FOR

Since WA is a Complex*16 variable, it is necessary that NIM * MIM < 64 * 1024 / 16 = 4096; and since UR is also a Complex*16 variable, it is necessary that JIM * MIM < 4096. If MIM = 300, then both NIM AND JIM must not exceed 13. If MIM is reduced to 200, then JIM and NIM must not exceed 20.

It is therefore recommended that, for PC-DOS use, the statement

```
PARAMETER (NIM = 20, MIM = 300, JIM = 101)
```

be amended to read

```
PARAMETER (NIM = 20, MIM = 200, JIM = 20)
```

RELINT.FOR

```
PARAMETER (KIM = 480)
DIMENSION UMP (MIM), RK(KIM), tloss(KIM,JIM)
```

Since TLOSS is a Real*4 variable, it is necessary that KIM * JIM < 64 * 1024 / 4 = 16384. If JIM = 20, KIM must be no more than 819; whereas if JIM = 13, KIM must be no more than 1260. In order to run on a PC-DOS, it is not necessary to reduce KIM unless JIM is set to be greater than 34.

2. Directory Structure

Since the directory structure on a PC will generally be different to that on the VAX, the following amendments should be made:

FNCTNS.FOR

Delete *[.plot]* in the statement:

```
open (unit = jbin, file = '[.plot]modefns.dat',
:  status = 'unknown', form = 'unformatted')
```

RELINT.FOR

Delete *[.plot]* in the statement:

```
open (unit = jput, file = '[.plot]tloss'//fname(13:14)//'.dat',
:  status = 'new', form = 'unformatted')
```

REPORT NO.	AR NO.	REPORT SECURITY CLASSIFICATION
MRL-CP-1	AR-008-263	Unclassified

TITLE

The STOKES (Version 4) normal-mode shallow water transmission loss program

AUTHOR(S)
Marshall V. Hall

CORPORATE AUTHOR
DSTO Materials Research Laboratory
PO Box 50
Ascot Vale Victoria 3032

REPORT DATE	TASK NO.	SPONSOR
June, 1993	89/252	

FILE NO.	REFERENCES	PAGES
G6/4/8-4398	-	58

CLASSIFICATION/LIMITATION REVIEW DATE

CLASSIFICATION/RELEASE AUTHORITY
Chief, Maritime Operations Division

SECONDARY DISTRIBUTION

Approved for public release

ANNOUNCEMENT

Announcement of this report is unlimited

KEYWORDS

ABSTRACT

The STOKES program calculates the Normal-Mode-Sum solution to the acoustic wave equation for an arbitrary sound-speed profile in a water column overlying a seabed that consists of a liquid layer and a solid half-space. The water column sound-speed profile may be arbitrary except that the sound-speed adjacent to the sea-floor must not be significantly greater than that at the depths of the sound transducers. For the seabed, the user can either supply values for each geoacoustic parameter, or if the seabed is unconsolidated sediment, supply the mean grain-size. In the latter case, STOKES uses published regression equations and models to estimate the geoacoustic parameters. The mode phase-speed eigenvalues are the zeroes of a characteristic function that depends on the depth function at the sea-floor and the reflectivity of the sea-floor. STOKES calculates the mode eigenvalues to great accuracy, and is superior to SNAP if the seabed has a significant shear-speed. STOKES has been successfully run for realistic cases with bottom depths of from 60 to 120 m, and for frequencies from 32 Hz to 2 kHz. It has also performed very well when compared with bench-marks. At very low frequencies, near the cut-off frequency of the sound-speed profile as a whole (usually of the order of 10 Hz in practical cases), the imaginary part of any eigenvalue is large, and in some cases this prevents STOKES, which uses a simple root-finding procedure, from finding any mode. In such a case the Transmission Loss (at ranges of at least a few kilometres) would be exceptionally high, and its precise value would not be of practical interest. .

The STOKES (Version 4) Normal-Mode, Shallow Water Transmission Loss Program

Marshall V. Hall

(MRL-CP-1)

DISTRIBUTION LIST

Director, MRL	
Chief, Maritime Operations Division	
Research Leader, Submarine and Ship Sonar	
Dr Marshall V. Hall	
Dr D.H. Caro	
Miss S. Prenc	
Research Leader, Airborne Sonar	
Dr A. Larsson	
Dr R. O'Dowd	
Research Leader, Maritime Operations Research	
Research Leader, Mine Warfare Signatures	
MRL Information Services	
Chief Defence Scientist (for CDS, FASSP, ASSCM)	1 copy only
Director, Surveillance Research Laboratory	
Director (for Library), Aeronautical Research Laboratory	
Director, Electronics Research Laboratory	
Head, Information Centre, Defence Intelligence Organisation	
OIC Technical Reports Centre, Defence Central Library	
Officer in Charge, Document Exchange Centre	8 copies
Army Scientific Adviser, Russell Offices- data sheet only	
Air Force Scientific Adviser, Russell Offices	
Navy Scientific Adviser	
Scientific Adviser, Defence Central	
Director General Force Development (Land) - data sheet only	
Senior Librarian, Main Library DSTOS	
Librarian - MRL Sydney	5 copies
Librarian, H Block	
UK/USA/CAN/ABCA Armies Standardisation Rep., c/- DATD (NSO)	3 copies
Librarian, Australian Defence Force Academy	
Counsellor, Defence Science, Embassy of Australia - data sheet only	
Counsellor, Defence Science, Australian High Commission - data sheet only	
Scientific Adviser to DSTC Malaysia, c/- Defence Adviser - data sheet only	
Scientific Adviser to MRDC Thailand, c/- Defence Attache - data sheet only	
Head of Staff, British Defence Research and Supply Staff (Australia)	
NASA Senior Scientific Representative in Australia	
INSPEC: Acquisitions Section Institution of Electrical Engineers	
Head Librarian, Australian Nuclear Science and Technology Organisation	
Senior Librarian, Hargrave Library, Monash University	
Library - Exchange Desk, National Institute of Standards and Technology, US	
Exchange Section, British Library Document Supply Centre	
Periodicals Recording Section, Science Reference and Information Service, UK	
Library, Chemical Abstracts Reference Service	
Engineering Societies Library, US	
Documents Librarian, The Center for Research Libraries, US	

DISTRIBUTION LIST
(Continued)

Director General Force Development (Sea), HQ ADF, Russell Offices (B-4-05)
Director Combat Force Development (Sea), HQ ADF, Russell Offices (B-4-04)
DDSMWD, HQ ADF, Russell Offices (B-4-03)
DDSWD, HQ ADF, Russell Offices (B-4-34A)
Director Above & Underwater Warfare, Navy Office, Russell Offices (A-3-19)
Director Submarine Policy and Warfare, Navy Office, Russell Offices (A-3-24)
Director Oceanography & Meteorology, Maritime Headquarters, Potts Point NSW 2011
Physics Department, Australian Defence Force Academy, Campbell ACT
Defence Scientific Establishment, Auckland Naval Base Post Office, Auckland, NZ
Head of Sonar Performance Group, Defence Research Agency, Portland Dorset DT5 2JS, UK
Dr D.M.F. Chapman, Defence Research Establishment Atlantic, PO Box 1012, Dartmouth, Nova Scotia B2Y 3Z7, Canada
Dr F.B. Jensen, SACLANT Undersea Research Centre, Viale San Bartolomeo 400, I-19026 La Spezia, Italy
Dr W.A. Kuperman, Code 5107, Naval Research Laboratory, Washington DC 20375-5000, USA
Dr H. Schmidt, Department of Ocean Engineering, Massachusetts Institute of Technology, Cambridge MA 02139, USA